



INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification: G06F 17/60	A1	(11) International Publication Number: WO 00/77709
		(43) International Publication Date: 21 December 2000 (21.12.2000)
(21) International Application Number: PCT/US00/16526		
(22) International Filing Date: 13 June 2000 (13.06.2000)		Published
(30) Priority Data: 60/139,113 14 June 1999 (14.06.1999) US 60/162,873 01 November 1999 (01.11.1999) US		
(60) Parent Application or Grant INTEGRAL DEVELOPMENT CORPORATION [/]; O. TOLAT, Viral, Vipin [/]; O. REES, Stephen [/]; O. SANDHU, Harpal, S. [/]; O. TOLAT, Viral, Vipin [/]; O. REES, Stephen [/]; O. SANDHU, Harpal, S. [/]; O. CHOU, Chien-Wei (Chris) ; () .		

(54) Title: SYSTEM AND METHOD FOR CONDUCTING WEB-BASED FINANCIAL TRANSACTIONS IN CAPITAL MARKETS
 (54) Titre: SYSTEME ET PROCEDE DESTINES A OPERER DES TRANSACTIONS FINANCIERES SUR LE MARCHE DES CAPITAUX VIA L'INTERNET

(57) Abstract

A system and method (Fig. 1) is provided to engage in capital market transactions via the Internet (10). Through a system of servers (20, 90), application (3280), and interfaces (3275), financial instrument trading (160), portfolio management (170), and financial analyses (190) are seamlessly performed. Automated communications (1070) enabling connectivity with user systems (1150) are facilitated using XML-based syntax (Fig. 10) and XSL-based programming language.

(57) Abrégé

La présente invention concerne un système et un procédé permettant d'opérer des transactions sur le marché des capitaux par l'intermédiaire d'Internet (10). Un système de serveurs (20, 90), d'applications (3280) et d'interfaces (3275) permet d'effectuer des échanges d'instruments financiers (160), de la gestion de portefeuilles (170) et des analyses financières (190) en continu. Les communications automatisées (1070) offrant une connectivité avec des systèmes utilisateur (1150) sont améliorées au moyen d'une syntaxe XML (Fig. 10) et d'un langage de programmation XSL.

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau



(43) International Publication Date
21 December 2000 (21.12.2000)

PCT

(10) International Publication Number
WO 00/77709 A1

(51) International Patent Classification⁷: G06F 17/60

(21) International Application Number: PCT/US00/16526

(22) International Filing Date: 13 June 2000 (13.06.2000)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/139,113 14 June 1999 (14.06.1999) US
60/162,873 1 November 1999 (01.11.1999) US

(71) Applicant (*for all designated States except US*): INTEGRAL DEVELOPMENT CORPORATION [US/US]; 2023 Stierlin Court, Mountain View, CA 94043 (US).

(72) Inventors; and

(75) Inventors/Applicants (*for US only*): TOLAT, Viral, Vipin [US/US]; 2148 Harkins Avenue, Menlo Park, CA 94025 (US). REES, Stephen [GB/GB]; Flat 2, The Oak, Knoll Hill, Sneyd Park, Bristol, BS9 1QU (GB). SANDHU, Harpal, S. [—US]; 669 Waverly Street, Palo Alto, CA 94301 (US).

(74) Agent: CHOU, Chien-Wei (Chris); Oppenheimer Wolff & Donnelly LLP, 1400 Page Mill Road, Palo Alto, CA 94304 (US).

(81) Designated States (*national*): AE, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CR, CU, CZ, DE, DK, DM, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZA, ZW.

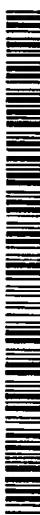
(84) Designated States (*regional*): ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:

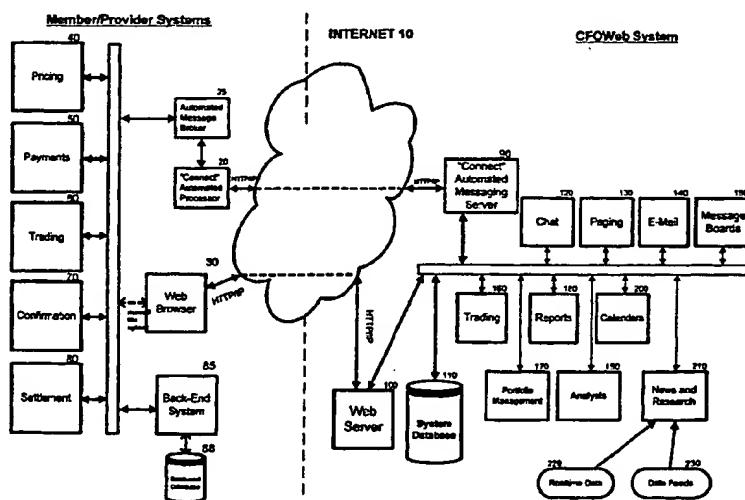
- With international search report.
- Before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments.

[Continued on next page]

(54) Title: SYSTEM AND METHOD FOR CONDUCTING WEB-BASED FINANCIAL TRANSACTIONS IN CAPITAL MARKETS



WO 00/77709 A1



(57) Abstract: A system and method (Fig. 1) is provided to engage in capital market transactions via the Internet (10). Through a system of servers (20, 90), application (3280), and interfaces (3275), financial instrument trading (160), portfolio-management (170), and financial analyses (190) are seamlessly performed. Automated communications (1070) enabling connectivity with user systems (1150) are facilitated using XML-based syntax (Fig. 10) and XSL-based programming language.

WO 00/77709 A1



For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

Description

5

10

15

20

25

30

35

40

45

50

55

5

**System and Method for Conducting Web-based
Financial Transactions in Capital Markets**

10

This is a continuation-in-part application of: (i) U.S. Provisional Patent Application Serial No. 60/139,113 filed June 14, 1999, entitled "SYSTEM AND METHOD FOR AN XML VOCABULARY FOR CAPITAL MARKETS"; and (ii) U.S. Provisional Patent Application Serial No. 60/162,873 filed November 1, 1999, entitled "METHOD AND APPARATUS FOR WEB-BASED MANAGEMENT OF FINANCIAL RISK AND PRICING AND TRADING OF FINANCIAL PRODUCTS". This application incorporates by reference U.S. Provisional Patent Application Nos. 60/139,113 and 60/162,873.

20

FIELD OF THE INVENTION

25

This invention relates generally to the field of interactive and automated Web-based financial transaction applications, and in particular to interactive and automated systems and methods for conducting financial transactions and managing portfolios and related financial information in capital markets.

30

BACKGROUND

35

During the evolution of the Internet, including the World Wide Web, there has been a continual introduction of applications and services to enable individuals and organizations to conduct financial research, manage their financial portfolios, and engage in certain types of financial transactions. The wide array of applications and services ranges from on-line banking to stock quote and financial information services to sites that enable users to engage in on-line, real-time market trades involving various instruments such as stocks, stock options, bonds, and mutual funds. The trading services, for example E*TRADE Securities, Inc.'s "E*TRADE" <www.etrade.com>, Charles Schwab & Co., Inc.'s "Schwab.com" <www.schwab.com>, and Fidelity Brokerage Services, Inc.'s "Fidelity.com" <www.fidelity.com>, permit trading of standard instruments in recognized markets. In such services, the investor uses credit or an account set up through the trade service to engage in trades through the service's proprietary system and interfaces. Such services, which are geared towards individual investors, do not permit seamless integration with users' internal or

40

45

50

55

WO 00/77709

5 back-end systems or the creation and trading of customized
many others like them, do not enable trading between parties in cu
foreign exchange, or the pricing and modeling of other capital market tra

Some steps have been taken to tap into the potentially vast market of in

10 5 investors wishing to engage in complex transactions via the Internet. The "Open Fin
Exchange" (Intuit Inc., Microsoft Corp., CheckFree Corp.) <www.ofx.net> was created to
provide a common specification for the electronic exchange of financial data between
15 financial institutions, businesses, and consumers via the Internet that enables financial data
exchange among disparate systems, in order to support online banking, bill payment and
10 presentment, and the trading of stocks, bonds, and mutual funds. The Open Financial
Exchange does not, however, provide a vocabulary, platform, and communication protocol to
enable the creation, negotiation, and execution of complex, capital market transactions among
20 financial institutions and institutional investors.

What is needed is a system and method that enables institutional investors and
25 15 financial institutions to seamlessly create, price, negotiate, execute, settle and analyze
complex, capital market transactions, including interest and currency derivatives, foreign
exchange, loans and deposits, and fixed-income instruments, using a standard vocabulary and
messaging system that enables seamless integration with the proprietary, existing systems of
30 the users.

20

SUMMARY

35 The present invention provides a system and method that enables users, such as
"Members" (e.g., institutional investors) and "Providers" (e.g., financial institutions), to
engage in capital market transactions, including the trading of Over-the-Counter financial
25 products, via the Internet (including the World Wide Web). The system includes a variety of
servers, applications, and interfaces that enable users to interactively communicate and trade
financial instruments among one another, and to manage their portfolios. Interactive
40 communications supported by the system include: requesting price quotes, monitoring and
reviewing quote requests, issuing price quotes, monitoring and reviewing price quotes,
45 30 negotiation between Members and Providers, acceptance of price quotes, reporting, portfolio
management, analysis of financial information and market data, calendaring, and

50

5 communications among Members, Providers, and/or system administrators, including e-mail, chat, and message boards.

The present invention also supports communications with the server side in an automated manner via an automated processor (the "Connect Processor" and "Connect
10 5 Messaging Server"). Such automated communications enable connectivity with users' internal, back-end systems to execute automated, straight-through processing, including transaction pricing, payment scheduling and journaling, derivatives trading, trade
15 confirmation, and trade settlement. Such communications are facilitated using a novel XML-based syntax ("FinXML") and XSL-based processing language ("FinScript"). FinXML
20 10 provides a standard data interchange language for capital market transactions and supports a broad set of elements and attributes that represent a wide variety of financial transactions, reference data, and market data. The common description of the FinXML syntax can be used for all aspects of straight-through-processing, including deal creation, confirmation, settlement, payment, risk management, and accounting.

25 15

BRIEF DESCRIPTION OF THE FIGURES

The above objects and description of the present invention may be better understood with the aid of the following text and accompanying drawings:
30

FIG. 1 shows the architecture of an embodiment of the present invention.

20 **FIG. 2** shows a flowchart of the process by which Members and Providers conduct a financial transaction in an embodiment of the present invention.

35 **FIG. 3** shows the structure of a FinXML "Trade" element in an embodiment of the present invention.

40 **FIG. 4** shows the structure of a FinXML "External Party" element in an embodiment of the present invention.

45 **FIG. 5** shows the structure of a FinXML "Internal Party" element in an embodiment of the present invention.

FIG. 6 shows the structure of a FinXML "Events" element in an embodiment of the present invention.

50 **FIG. 7** shows the general architecture of the Connect Automated Processor in an embodiment of the present invention.

5 **FIG. 8** shows an architectural overview of the Connect Automated Processor in an embodiment of the present invention.

10 **FIG. 9** shows the layout of a Connect Message in an embodiment of the present invention.

15 **FIG. 10** shows the structure of a Connect Message in an embodiment of the present invention.

15 **FIG. 11** shows a diagram of the Connect Message Flow for the automated pricing (synchronous) function in an embodiment of the present invention.

10 **FIG. 12** shows a diagram of the Connect Message Flow for the automated pricing (asynchronous) function in an embodiment of the present invention.

20 **FIG. 13** shows a diagram of the Connect Message Flow for the semi-automated pricing (synchronous) function in an embodiment of the present invention.

25 **FIG. 14** shows a diagram of the Connect Message Flow for the deal transmission (asynchronous) function in an embodiment of the present invention.

15 **FIG. 15** shows the components utilized in converting financial objects into a FinXML document using FinScript in an embodiment of the present invention.

30 **FIG. 16** shows a flowchart of the process of converting financial objects into a FinXML document using FinScript in an embodiment of the present invention.

20 **FIG. 17** shows the components utilized in converting a FinXML document into financial objects using FinScript in an embodiment of the present invention.

35 **FIG. 18** shows a flowchart of the process of converting a FinXML document into financial objects using FinScript in an embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

40 25 A. **System Functionality**

The technology of the present invention can be embodied in various forms to provide a platform for conducting interactive and automated financial transactions and management of portfolios and related financial information in capital markets. The platform enables members, including end-users and providers of financial products and services, to engage in the trading of standard and customized financial instruments in capital markets. System functionality includes: capture and pricing of financial instrument trades; presentation of real-time market data; saving of completed trades to a portfolio; management of trading

5 workflow; transmission of trades to end-users' proprietary, back-end systems for pricing, trading, payment processing, confirmation, and settlement; performance of portfolio analysis; performance of risk management analysis; and inter-user communications.

10 In the present embodiment of the invention, the system includes both server-side and
5 client-side functionality. The server-side functionality enables system users to interactively and seamlessly: engage in financial instrument trades; perform portfolio management, analysis, and reporting; obtain real-time market data and news; communicate with the system and other users via electronic mail, chat, and message boards; and maintain a calendar. The server-side includes interactive system servers that host such user activities, as well as one or
15 more system databases, and an automated messaging server that controls communication with the automated back-end systems of clients.

20 The client side functionality includes an automated processor that communicates with the automated messaging server of the system side and serves as a seamless interface to the automated back-end systems and proprietary databases of clients. Thus, the system enables
25 organizations with disparate systems and data to engage in transactions using the common
15 functionality and interfaces of this invention. The client side also includes client web browsers that enable interactive communication with the system servers.

30 The invention described herein provides a standard, XML-based vocabulary to represent and facilitate the financial transactions, as well as a system and method for
20 converting users' data and information to/from the standard vocabulary and communicating such information through the system in an automated manner.

35 1. System Architecture Overview

40 FIG. 1 illustrates the architecture of one embodiment of this invention. This
25 embodiment is presented for purposes of illustration and description, and other embodiments will be apparent to and could be implemented by practitioners skilled in this art.

45 a. Server Side

50 The server side (sometimes referred to as the "CFOWeb System" in this embodiment) communicates with the client side (consisting of users known as "Members", e.g., corporations and institutional investors, and "Providers", e.g., financial institutions) via the Internet (including the World Wide Web) 10. The server side includes a variety of

5 interactive system servers that provide functionality to users. Web server 100 enables
communications (through the Internet via a transfer protocol such as, e.g., HyperText
10 Transfer Protocol or "HTTP/IP") between users who connect to the server side through their
web browsers 30 and the various system servers. Trading server 160 provides a graphic user
15 interface and applications that enable users to interactively trade financial derivatives among
each other. Portfolio management server 170 provides a graphic user interface and
applications that enable users to manage their portfolios of financial derivatives. Reports
server 180 provides a graphic user interface and applications that enable users to run and
produce standard and customizable reports regarding their portfolios, including mark to
20 market, upcoming events, and trade lists. Analysis server 190 provides a graphic user
interface and applications that enable users to run analytics against their portfolios, including
valuation, and interest rate sensitivity. Calendar server 200 provides a graphic user interface
and applications that enable users to automatically calendar key dates regarding settlement,
payments, cash flows, and other details related to their financial derivative transactions and
25 portfolios. News and research server 210 provides a graphic user interface and applications
that enable users to obtain real-time market data and financial and other news, as well as
proprietary third-party data feeds. News and research server 210 includes connections to real-
time market data feeds and news services 220 and third-party data feeds 230.

30 The interactive system servers also include servers that enable communication among
20 system users and administrators. Chat server 120 provides real-time chat, thus enabling users
to engage in discussion forums related to financial derivatives. Paging server 130 enables
35 users to build a messaging community and determine which users are online and available to
receive messages at a given instance. E-mail server 140 provides an intra-system electronic
mail vehicle, enabling communications among users and system administrators, including all
40 aspects of a financial trade from quote requests to settlement. Message boards server 150
provides an arena for users and system administrators to post and read system-wide messages,
as well as quote requests and quotes.

45 Automated messaging server 90 (sometimes referred to as the "Connect Messaging
Server" in this embodiment) controls communications (through the Internet via a transfer
30 protocol, e.g., HTTP/IP) between the various system servers of the server side and users
whose internal, back-end systems 85 execute automated processes that require
50 communication with the server side. Such automated processes could include transaction

5 pricing 40, payment scheduling and journaling 50, derivatives trading 60, trade confirmation
10 70, and trade settlement 80. Communications between Connect Messaging Server 90 and the
client side pass through automated processor 20 (sometimes referred to as the "Connect
Processor" in this embodiment) – which shares the same functionality as automated
15 5 messaging server 90 – and automated message broker 25 and are facilitated using the
"FinXML" vocabulary and the "FinScript" processing language, as described below.

10 The CFOWeb System includes one or more system databases 110, which store data
15 related to the processing of financial transactions, as well as user communications and
interactions with the system servers.

10

20 **b. Client Side**

25 The client side includes functionality that enables users – Members and Providers –
30 to communicate, either interactively or in an automated manner, with the various system
servers. Web browser 30 enables interactive communications (through the Internet via a
35 transfer protocol, e.g., HTTP/IP) between users and the CFOWeb System with connection
made on the server side at web server 100. Interactive communications might include:
requesting price quotes (Members), monitoring and reviewing quote requests (Providers),
issuing price quotes (Providers), monitoring and reviewing price quotes (Members),
negotiation between Members and Providers, acceptance of price quotes (Members),
20 reporting, portfolio management, analysis of financial information and market data,
calendaring, and communications among Members, Providers, and/or system administrators,
including e-mail, chat, and message boards.

25 Alternatively, users can communicate with the server side in an automated manner via
30 Connect Processor 20 (and automated message broker 25) which communicates (through the
35 Internet via a transfer protocol, e.g., HTTP/IP) with Connect Messaging Server 90. Such
40 automated communications enable users' internal back-end systems 85 (which include one or
more back-end databases 88) to execute automated processes, which could include
transaction pricing 40, payment scheduling and journaling 50, derivatives trading 60, trade
confirmation 70, and trade settlement 80. Such communications are facilitated using the
45 30 "FinXML" vocabulary and the "FinScript" processing language, as described below.

50

5 **2. Financial Transaction Functionality**

For system users – Members and Providers – the functionality included in an embodiment of this invention can be categorized as follows: pre-transaction, transaction, post-transaction, and general. The present invention (i) automates or (ii) provides an
10 5 interactive interface for such functionality.

15 **a. Pre-Transaction**

Members and Providers (or in other embodiments of this invention, Members and Members) that engage in a financial transaction of a type enabled by this invention proceed through a series of steps illustrated in FIG. 2. When a Member and Provider decide that they may engage in financial transactions in the future, the parties establish their relationship by executing certain agreements (step 300). Such agreements govern the rules of engagement, rate sources, confirmation and settlement procedures, and other information that can be reused over a series of transaction between the parties. The International Swaps and
20 25 Derivatives Association, Inc. (“ISDA”) <<http://www.isda.org>> provides certain standardized agreements (e.g., “1992 ISDA Master Agreement”) that may be used by the parties for these purposes. The parties can carry out this step using the interactive e-mail function of the system (provided by e-mail server 140 in FIG. 1) to exchange information to be included in the agreements. In addition, by combining off-the-shelf electronic signature software with
30 35 the system, the parties can electronically sign and exchange the standardized agreements. Members and Providers may engage in multiple iterations of this step, depending upon the number of standardized agreements that the parties will execute.

Next, the Member and Provider will negotiate one or more lines of credit to be assigned by the Provider to the Member for future transactions (step 310). Each Member will negotiate such line(s) of credit with each Provider with which the Member intends to engage in future transactions. In assigning a credit line to a Member, the Provider will analyze the Member’s asset portfolio, credit ratings, and type of financial transactions to be executed by Member. The parties can carry out this step using the interactive e-mail function of the system (provided by e-mail server 140 in FIG. 1) to exchange information during the credit
40 45 line negotiations.

b. Transaction

Once the Member and Provider have established their relationship and negotiated a credit line, the Member can commence the process of engaging in a financial transaction.

The Member must decide on the type of transaction it wishes to execute (e.g., Foreign

Exchange Spot, Foreign Exchange Forward, Interest Rate Swap, etc.) and structure the

desired transaction (step 320). In this step, the Member will use the interactive trading

function of the system (provided by trading server 160 in FIG. 1), including graphic user

interfaces and tools. Depending upon the type of transaction, the structure might include

pricing variables, an expiration period, a list of Providers to whom the Member would like

request pricing, and any other particulars specific to the Member and the desired transaction.

For example, a Member might specify a Foreign Exchange Spot transaction in which the

Member desires to buy 1 million Euro currency for U.S. Dollars, with the transaction request

set to expire on July 30, 2000 at 11:59 P.M. Pacific Standard Time.

After structuring the transaction, the Member submits a request for pricing of the

transaction to one or more Providers (step 350), using the interactive trading function of the

system (provided by trading server 100 in FIG. 1). Alternatively, the Member might

function of the system ($\epsilon = 0.11$, $\mu = -3$, $\gamma = 140$; FIG. 1). Such a small

Communication would include a URL to the structured transaction and one or more files.

Providers monitor and review the Member's pricing request (step 340) via

communications between the automated messaging server 90 and automated processor 20

will be described below. Such communications result in the posting of pricing requests on a

request-monitoring interface hosted by the system. Upon reviewing the transaction and

pricing request, including information about the particular Member (unless the Member's

identity was not disclosed), a Provider can submit a pricing offer (i.e., price quote) to the

requesting Member (step 350). The submission of the pricing offer occurs via a

communication between the automated processor 20 and automated messaging server 90, as

will be described below. Each pricing offer typically has an expiration period because of

constantly changing market conditions, and the Provider may submit modified pricing offers

to the Member.

The Member can monitor and review any pricing offers submitted by Providers (step

5 pricing offers (step 370) and negotiate with the particular Provider(s) who provided the offer(s). In the present embodiment, such negotiations may occur using the interactive e-mail function of the system (provided by e-mail server 140 in FIG. 1) or through traditional methods (e.g., telephone calls). The number of iterations of negotiations will depend upon
10 5 the market volatility and other conditions. In other embodiments of this invention, such negotiations may be unnecessary if certain parameters are met by a Provider's pricing offer.

15 Following negotiations regarding pricing offers, the Member will accept the best pricing offer (step 380) and communicate its acceptance to the Provider using the interactive trading function of the system (provided by trading server 160 in FIG. 1). The Provider will
20 10 receive the Member's acceptance via communications between the automated messaging server 90 and automated processor 20, as will be described below. Such communications result in the posting of the Member's acceptance of the pricing offer on the request monitoring interface hosted by the system.

25 15 **c. Post-Transaction**

30 Upon receipt of the Member's acceptance, the Provider sends confirmation of the transaction to the Member (step 390), including specific terms, payment dates, and amounts. The Provider sends the confirmation information to the Member via communications between
35 20 the automated processor 20 and the automated messaging server 90, as will be described below. The Provider's back-end system 85 provides automated processing of this information.

40 25 Following confirmation, the Member and Provider will submit the transaction to their respective back-end systems 85 (step 400) for purposes including internal accounting and payment scheduling. This step can be handled by the system via an automated connection between the automated processor 20 and the back-end system 85. Using their respective
45 25 back-end systems 85, the Member and Provider schedule settlement of the transaction and future cash flows (step 410).

45 **d. General**

50 30 Interactive system functionality that can be accessed and implemented at any time by the Member and Provider includes: reporting; portfolio management; risk management; analysis of financial information and market data; e-mail communication with Members,

5 Providers, and system administrators; chat with Members and Providers; message boards; calendaring; and paging.

B. Automated Processing and Transferring of Financial Information

10 The present embodiment of this invention supports financial transactions between Members and Providers by providing automated processing and transfer of the underlying financial information between the messaging server of the server side and the automated processor of the client side. The system enables such processing and transfer by using a novel tag-based language – FinXML™ – that describes financial instrument trades, including
15 transaction-specific data, reference data, and market data. FinXML conforms to the
10 Extensible Markup Language (XML) 1.0 Recommendation (Feb. 10, 1998), World Wide
20 Web Consortium (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University) <<http://www.w3.org/TR/REC-xml>>. The XML Recommendation describes a set of rules for conforming documents that is based
25 around the use of element tags which mark the components of a document or describe the structure of data files as textual documents.

30 FinXML also conforms to the 1991 ISDA Definitions (and 1998 Supplement) of the International Swaps and Derivatives Association, Inc. (“ISDA”) <<http://www.isda.org>>. The ISDA Definitions provide a set of standard terms for use in privately-negotiated financial
20 derivatives transactions. The element tags and attribute names and values defined in FinXML, as described below, correspond to the terms defined in the ISDA Definitions.

35 FinXML, as a type of XML vocabulary, is ideally suited to electronic transmission over corporate intranets, extranets, and the Internet (including the World Wide Web), using a
transfer protocol such as HTTP/IP. HTTP/IP is intended to transmit text documents such as
25 the HyperText Markup Language (“HTML”) documents used to describe the pages to be displayed in a Web browser. XML documents – and, thus, FinXML documents – are similar to HTML documents in that both types of documents are text-based, both consist of a mixture of element tags and data content, and both may include references to other external material.

40 In a basic financial transaction between two organizations, a financial transaction encoded in XML is sent using a transfer protocol such as HTTP/IP from a client application of one organization to a server of the other organization. The server, in turn, sends back a response that is also encoded in XML.

As will be described below, the present embodiment of this invention includes a novel method of encoding/decoding financial objects to/from FinXML (or other XML) documents using the automated processor 20 (also known as "Connect Processor") and automated messaging server 90 (also known as "Connect Messaging Server"). In a financial transaction between two organizations, one organization (e.g., a Member) submits a Java object to automated processor 20 which, as will be described below, uses a XML mapping and FinScript™ – proprietary stylesheets created in Extensible Stylesheet Language ("XSL") – to create a FinXML (or other XML) document that can be sent using a transfer protocol such as HTTP/IP to the automated messaging server 90 for conversion to an object and processing on the server side. Following processing, the automated messaging server 90 converts objects to a FinXML (or other XML) document and sends the document to the automated processor 20 which, as will be described below, uses FinScript to create a JavaScript program from the FinXML (or other XML) document. In turn, Java objects are created from the JavaScript program and sent to the other organization (e.g., a Provider). XSL, which serves as the foundation for FinScript, is described in the Extensible Stylesheet Language (XSL) Version 1.0 (March 27, 2000), World Wide Web Consortium (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University) <<http://www.w3.org/TR/xsl>>.

20 I. FinXML

In the present embodiment of this invention, FinXML documents are distributed between servers in order to communicate the details of financial transactions and related data. The FinXML syntax provides a general structure for all financial transactions. The financial transactions, in turn, consist of underlying elements, each of which contains attributes and/or other elements.

a. Trade Structure

The basic financial transaction element of the FinXML syntax is a "Trade", of which there are multiple types (described below). The Trade element is the root element for the description of each financial transaction object. The Trade element is contained within the Connect message "payload" component (described below).

5 FIG. 3 illustrates the structure of a Trade element. Trade element 500 contains at
least one pair of "Counterparty" elements 510, which are the parties engaged in the
transaction. Each Counterparty element 510 can be an "Internal Party" element 515 or an
10 "External Party" element 520 (described below). Trade element 500 also contains a "Trade
5 Type" element 530, which contains one of the following Trade Type sub-elements:

- (1) Foreign Exchange ("FX") Spot
- (2) FX Forward
- (3) Interest Rate Fixed Float Swap
- (4) Interest Rate Float Float Swap
- 10 (5) Cap
- (6) Floor
- (7) Fixed Deposit
- (8) Fixed Loan
- (9) Float Deposit
- 20 (10) Float Loan
- (11) FX Option
- (12) FX Swap
- 30 (13) Cross Currency Fixed Fixed Swap
- (14) Cross Currency Fixed Float Swap
- 20 (15) Cross Currency Float Float Swap
- (16) Forward Rate Agreement
- 35 (17) Customized Trade

Each Trade Type element represents a different type of financial transaction, which will be
described separately below.

40 In the present embodiment of this invention, Trade element 500 has the following
XML definition:

```
<!ELEMENT trade (%parties; . (fxSpot | fxForward | interestRateFixedFloatSwap |  
interestRateFloatFloatSwap | cap | floor | fixedDeposit | fixedLoan | floatDeposit | floatLoan | fxOption | fxSwap  
| crossCurrencyFixedFixedSwap | crossCurrencyFixedFloatSwap | crossCurrencyFloatFloatSwap |  
forwardRateAgreement | customizedTrade ) )>  
<!ATTLIST trade tradeId CDATA #REQUIRED isBuiltFromParameters CDATA #IMPLIED >
```

5

b. Financial Transaction Data

The FinXML syntax describes various types of data that comprise a financial transaction, including transaction data, reference data, and market data. Each of these types of data includes elements and attributes.

10

i. Transaction Data

15

Transaction data describes the various components of a financial transaction or trade. These components include "Counterparty" elements, "Trade Type" elements, "Trade Specific" elements, "Financial Event" elements, and "Calculation" elements.

20

(a) Counterparty Elements

25

In a financial transaction of the type described by FinXML, there are typically two parties, also referred to as "counterparties". As described above, FinXML describes such parties to a transaction with Counterparty element 510 (as shown in FIG. 3), including an Internal Party element and an External Party element. In the present embodiment of this invention, Counterparty element 510 has the following XML definition:

30

```
<!ENTITY % counterParty "internalParty | externalParty">
<!ENTITY % parties "(&%counterParty;); (&%counterParty;)">
```

20

In each transaction, from the perspective of an organization, that organization is the "internal" party and the other, unrelated organization is the "external" party, e.g., a corporation and a third-party bank that engages in a foreign exchange transaction. Alternatively, where a corporation engages in a transaction with a subsidiary legal entity within the corporation, the subsidiary is also an "internal" party.

40

FIG. 4 illustrates the structure of the External Party element 700, which represents an external party to a transaction. Each external party can be either a "disclosed party" or an "undisclosed party". In the present embodiment of this invention, External Party element 700 has the following XML definition:

45

```
<!ELEMENT externalParty ((disclosedParty | undisclosedParty))>
<!ATTLIST externalParty id ID #IMPLIED type CDATA #IMPLIED >
```

50

55

5 Disclosed Party element 705 represents a party to a transaction (e.g., a Member) whose details, including corporate identification, are fully known to the other party to the transaction. Each Disclosed Party element 705 includes the following sub-elements (described in greater detail below in the discussion regarding "Reference Data" elements):

10 5 Organization 710, Contact Information 730, Address 765, and Credit Rating 805. In the present embodiment of this invention, Disclosed Party element 705 has the following XML definition:

15 <!ELEMENT disclosedParty (organization, contactInformation*, address, creditRating+)>

20 10 Undisclosed Party element 835 represents a party that remains anonymous to the other party; the only information disclosed is the party's credit rating. Thus, each Undisclosed Party element 835 includes the Credit Rating 805 element (described in greater detail below in the discussion regarding "Reference Data" elements). In the present embodiment of this invention, Undisclosed Party element 805 has the following XML definition:

25 15 <!ELEMENT undisclosedParty (creditRating+)>

30 FIG. 5 illustrates the structure of the Internal Party element 600, which represents an internal party to a transaction. Internal Party element 600 includes Legal Entity element 605, which represents each of the separate legal (i.e., corporate) entities associated with the 35 20 internal party, and Book element 625, which represents the trading book(s) in which a party will group transactions for accounting purposes (described in greater detail below in the discussion regarding "Reference Data" elements). In the present embodiment of this invention, Internal Party element 600 has the following XML definition:

40 25 <!ELEMENT internalParty (legalEntity?, book?)>

<!ATTLIST internalParty id ID #IMPLIED type CDATA #IMPLIED >

45 (b) Trade Type Elements

30 50 As shown in FIG. 3, Trade element 500 includes Trade Type element 530. Each Trade Type element 530, in turn, includes a Trade Type sub-element that describes one type of financial transaction or trade.

5

(1) Foreign Exchange Spot

A Foreign Exchange Spot ("FX Spot") transaction is one in which one party acquires a specified quantity of one currency in exchange for another currency from another party, to be paid or settled as soon as is standard (*i.e.*, usually two days) in the foreign exchange market. For example, a Member buys from a Provider 2 million Euro for U.S. Dollars to be paid in two days.

15 The FX Spot element represents such a transaction and includes the following sub-elements and attributes:

- 10 • "Dealt Amount": the specified amount of currency to be converted into the currency being acquired.
- 15 • "Settled Amount": the amount of currency being acquired.
- 20 • "Trade Date": the date on which the currency trade has been agreed to by the parties.
- 25 • "Value Date": the date on which the traded currencies will be exchanged (*i.e.*, the trade will be settled).
- 30 • "FX Rate": the foreign exchange rate at which the trade will be executed.
- 35 • "Base Currency": the currency against which the currency to be acquired will be measured.
- 40 • "Base Units": the number of units of the Base Currency against which the currency to be acquired will be quoted (usually one unit).
- 45 • "Quote Currency": the currency to be acquired or the currency to which the quote is pegged.
- 50 • "Quote Units": the number of units of the Quote Currency to be acquired.

5

In the present embodiment of this invention, the FX Spot element has the following XML definition:

```

10      5      <!-- Foreign Exchange Trades - FXSPOT -->
<!ENTITY % fxTradeSpec "%trade.elements;
           , dealtAmount
           , settledAmount">
15      10     <!ELEMENT fxSpot (%fxTradeSpec; )>
<!ELEMENT dealtAmount (currency, amount )>
<!ATTLIST dealtAmount %payReceiver; >
<!ELEMENT settledAmount (currency, (fxRate | amount ) )>
<!ATTLIST settledAmount %payReceiver; >
20      15     <!ELEMENT fxRate (baseCurrency, baseUnits, quoteCurrency, quoteUnits. rate )>
<!ELEMENT baseCurrency (#PCDATA )>
<!ELEMENT baseUnits (#PCDATA )>
25      15     <!ELEMENT quoteCurrency (#PCDATA )>
<!ELEMENT quoteUnits (#PCDATA )>
<!ENTITY % trade.elements "tradeDate
           , settlementDate?
           , valueDate?
           , externalId?">
30      20

```

(2) Foreign Exchange Forward

35 25 A Foreign Exchange Forward ("FX Forward") transaction is one in which one party acquires a quantity of one currency in exchange for a specified amount of another currency from another party, with the amounts to be paid on a specified future date. For example, a Member buys from a Provider 2 million Euro for U.S. Dollars to be paid 60 days from the trade date.

40 30 The FX Forward element represents such a transaction and includes the following sub-elements and attributes:

- 45 • "Dealt Amount": the specified amount of currency to be converted into the currency being acquired.

50

55

- 5 • “Settled Amount”: the amount of currency being acquired.

- 10 5 • “Trade Date”: the date on which the currency trade has been agreed to by the parties.

- 15 10 • “Value Date”: the date on which the traded currencies will be exchanged (*i.e.*, the trade will be settled).

- 20 15 • “FX Rate”: the foreign exchange rate at which the trade will be executed.

- 25 20 • “Base Currency”: the currency against which the currency to be acquired will be measured.

- 30 25 15 • “Base Units”: the number of units of the Base Currency against which the currency to be acquired will be quoted (usually one unit).

- 35 30 20 • “Quote Currency”: the currency to be acquired or the currency to which the quote is pegged.

- 40 35 25 • “Quote Units”: the number of units of the Quote Currency to be acquired.

45 In the present embodiment of this invention, the FX Forward element has the following XML definition:

```

5      <!-- Foreign Exchange Trades - FXFORWARD -->
<!ENTITY % fxTradeSpec "%trade.elements;
           , dealtAmount
           , settledAmount">
10     5      <!ELEMENT fxForward (%fxTradeSpec; )>
<!ELEMENT dealtAmount (currency, amount )>
<!ATTLIST dealtAmount %payReceiver; >
<!ELEMENT settledAmount (currency, (fxRate | amount ) )>
<!ATTLIST settledAmount %payReceiver; >
15     10     <!ELEMENT fxRate (baseCurrency, baseUnits, quoteCurrency, quoteUnits, rate )>
<!ELEMENT baseCurrency (#PCDATA )>
<!ELEMENT baseUnits (#PCDATA )>
<!ELEMENT quoteCurrency (#PCDATA )>
20     15     <!ELEMENT quoteUnits (#PCDATA )>
<!ENTITY % trade.elements "tradeDate
           , settlementDate?
           , valueDate?
           , externalId?">
25

```

20

30

(3) Interest Rate Fixed-Float Swap

An Interest Rate Fixed-Float Swap is a type of interest rate swap in which two parties exchange periodic payment streams, where one payment stream is based on a fixed interest rate and the other payment stream is based on a floating rate index (e.g., LIBOR), with each payment stream in the same currency. For example, a Member buys from a Provider a fixed payment stream in Euro in exchange for a floating payment stream in Euro based on the LIBOR index.

The Interest Rate Fixed-Float Swap element represents such a transaction and includes the following sub-elements and attributes:

- “Trade Date”: the date on which the trade has been agreed to by the parties.
- “Start Date”: the date on which the exchanged payments will begin.

50

- 5 • “End Date”: the date on which the exchanged payments will end.
- 10 5 • “Fixed Leg Details”: the details of the fixed interest payments for the fixed leg.
- 15 10 • “Float Leg Details”: the details of the floating interest payments for the floating leg.
- 20 15 • “Events”: the various payment and calculation events in the swap transaction, including cash payment, principal payment, interest payment, interest calculation, compound interest calculation, and interest rate reset information.

20 In the present embodiment of this invention, the Interest Rate Fixed-Float Swap element has the following XML definition:

```
<!-- Interest Rate Fixed Float Swap -->
15 <!ELEMENT interestRateFixedFloatSwap (tradeDate, startDate, endDate, externalId?, fixedLegDetails,
floatLegDetails, events?)>
```

30 (4) **Interest Rate Float-Float Swap**

20 An Interest Rate Float-Float Swap is a type of interest rate swap in which two parties exchange periodic payment streams, where each payment stream is based on a floating rate index (e.g., LIBOR), with each payment stream in the same currency. For example, a Member buys from a Provider a floating payment stream in Euro in exchange for a floating payment stream in Euro, where each payment stream is based on the LIBOR index.

25 The Interest Rate Float-Float Swap element represents such a transaction and includes the following sub-elements and attributes:

- “Trade Date”: the date on which the trade has been agreed to by the parties.
- “Start Date”: the date on which the exchanged payments will begin.
- “End Date”: the date on which the exchanged payments will end.

- 5 • “Float Leg Details”: the details of the floating interest payments; separate information for each of the two floating legs.
- 10 • “Events”: the various payment and calculation events in the swap transaction, including cash payment, principal payment, interest payment, interest calculation, compound interest calculation, and interest rate reset information.

15 In the present embodiment of this invention, the Interest Rate Float-Float Swap element has the following XML definition:

10 <!-- Interest Rate Float Float Swap -->
15 <!ELEMENT interestRateFloatFloatSwap (tradeDate, startDate, endDate, externalId?, floatLegDetails,
20 floatLegDetails, events?)>

15

25 (5) Cap

30 A Cap transaction is one in which one party, in exchange for a premium payment, acquires from another party the right to receive a payment stream from the other party with respect to a specified quantity of one currency if, on the scheduled payment dates, the level of
35 20 a specified rate or index exceeds an agreed “strike rate” for the period involved. For example, a Member purchases from a Provider an interest rate cap at a strike rate of 8 percent on U.S. Dollars based on the 3-month LIBOR for a period of 12 months, in order to hedge its exposure to increasing interest rates on a 10 million U.S. Dollars floating-rate loan based on the 3-month LIBOR.

40 25 The Cap element represents such a transaction and includes the following sub-elements and attributes:

- 45 30 • “Cap Floor Spec”: describes the structured elements common to Cap and Floor transactions.

- 50 35 • “Trade Date”: the date on which the trade has been agreed to by the parties.

- 5 • “Start Date”: the beginning date of the period for which the interest rate is
protected.
- 10 5 • “End Date”: the date on which the payment stream will end.
- 15 10 • “Premium Details”: the details of the premium to be paid, as either a percentage (“Premium Percentage”) or a specified amount (“Premium Amount”), and the payment date (“Premium Date”).
- 20 15 • “Strike Rate”: the rate that, if exceeded, will trigger the settlement of the transaction.
- 25 15 • “Buyer”: the buyer of the option to be exercised; this is a reference to a Counterparty element.
- 30 20 • “Writer”: the recipient of the premium for the option to be exercised; this is a reference to a Counterparty element.
- 35 25 • “Volatility Spread”: the spread over the volatility calculated using the volatility surface; an additional spread for pricing the cap transaction.
- 40 25 • “Discount Curve”: the definition of the discount curve used to calculate the payment stream.
- 45 30 • “Forecast Curve”: the definition of the forecast curve used to calculate the payment stream.
- 50 30 • “Notional Amount”: the amount used as the basis for calculating the payment stream.
- 55 50 • “Floating Interest Rate”: the floating interest rate.

5

- “First Fixing Rate”: the interest rate to be used for the first interest calculation period.

10

- 5 • “Day Count”: the day-count method to be used for calculating interest.
- “Payment Frequency”: the frequency of interest/principal payment.

15

- 10 • “Roll Date”: the specific day each month to be used for payment/settlement of interest/principal.

20

- “Payment Calendar”: the calendar to be used for reference to business holidays.
- “Rate Reset Calendar”: the calendar to be used for reference to business holidays for interest rate resets.
- “Date Stub”: an indicator for an irregular schedule of payments.

25

- 20 • “Anchor Date”: the date to which the payment schedule is anchored, i.e., the end date of the first interest period or specific date of first payment; could be the start of the last interest period if dates generated in reverse.

30

- 35 • “Amortization Details”: details regarding how the payment cashflow should be amortized, including amortization method (e.g., single payment at end, equal payments over term of stream).
- 40 • “Compounding Details”: details regarding how the interest should be compounded, including calculation frequency and rate.

45

- 30 In the present embodiment of this invention, the Cap element has the following XML definition:

50

```
5      <!-- Cap -->
6      <!ENTITY % capFloorSpec "premium details
7          . strikeRate
8          . volatilitySpread
9          . discountCurve?
10         . forecastCurve?">
11
12      <!ELEMENT cap (tradeDate, settlementDate?, startDate, endDate, externalID?,
13          %genericSpecDetails; , %floatRateDetails; , %capFloorSpec; , events? )>
14
15      <!ATTLIST cap buyer IDREF #REQUIRED writer IDREF #REQUIRED>
16      <!ELEMENT premiumDetails ( (premiumPercentage | premiumAmount) . premiumDate)>
17
18      <!ELEMENT premiumAmount (%currencyAmount; )>
19      <!ATTLIST premiumAmount %payReceiverAmount;>
20
21      <!ELEMENT premiumPercentage (#PCDATA )*>
22
23      <!ATTLIST premiumPercentage %payReceiverAmount;>
24
25      <!ELEMENT volatilitySpread (#PCDATA )>
26
27      <!ELEMENT discountCurve (#PCDATA )>
28
29      <!ELEMENT forecastCurve (#PCDATA )>
```

20

(6) Floor

A Floor transaction is one in which one party, in exchange for a premium payment, acquires from another party the right to receive a payment stream from the other party with respect to a specified quantity of one currency if, on the scheduled payment dates, the level of a specified rate or index is less than an agreed “strike rate” for the period involved. For example, a Member purchases from a Provider an interest rate floor at a strike floor level of 8 percent on U.S. Dollars based on the 3-month LIBOR for a period of 12 months, in order to protect its investment returns on a 10 million U.S. Dollars money market investment based on the 3-month LIBOR.

30 The Floor element represents such a transaction and includes the following sub-elements and attributes:

- “Cap Floor Spec”: describes the structured elements common to Cap and Floor transactions.

- 5 • “Trade Date”: the date on which the trade has been agreed to by the parties.
- 10 • “Settlement Date”: the date on which the trade will be settled.
- 15 • “Start Date”: the beginning date of the period for which the interest rate is
protected.
- 20 • “End Date”: the date on which the payment stream will end.
- 25 • “Premium Details”: the details of the premium to be paid, as either a percentage
 (“Premium Percentage”) or a specified amount (“Premium Amount”), and the
 payment date (“Premium Date”).
- 30 • “Strike Rate”: the rate that, if exceeded, will trigger the settlement of the
 transaction.
- 35 • “Buyer”: the buyer of the option to be exercised; this is a reference to a
 Counterparty element.
- 40 • “Writer”: the recipient of the premium for the option to be exercised; this is a
 reference to a Counterparty element.
- 45 • “Volatility Spread”: the spread over the volatility calculated using the volatility
 surface; an additional spread for pricing the cap transaction.
- 50 • “Discount Curve”: the definition of the discount curve used to calculate the
 payment stream.
- 55 • “Forecast Curve”: the definition of the forecast curve used to calculate the
 payment stream.

- 5 • “Notional Amount”: the amount used as the basis for calculating the payment stream.
- 10 • “Floating Interest Rate”: the floating interest rate.
- 15 • “First Fixing Rate”: the interest rate to be used for the first interest calculation period.
- 20 • “Day Count”: the day-count method to be used for calculating interest.
- 25 • “Payment Frequency”: the frequency of interest/principal payment.
- 30 • “Roll Date”: the specific day each month to be used for payment/settlement of interest/principal.
- 35 • “Payment Calendar”: the calendar to be used for reference to business holidays.
- 40 • “Rate Reset Calendar”: the calendar to be used for reference to business holidays for interest rate resets.
- 45 • “Date Stub”: an indicator for an irregular schedule of payments.
- 50 • “Anchor Date”: the date to which the payment schedule is anchored, *i.e.*, the end date of the first interest period or specific date of first payment; could be the start of the last interest period if dates generated in reverse.
- 55 • “Amortization Details”: details regarding how the payment cashflow should be amortized, including amortization method (*e.g.*, single payment at end, equal payments over term of stream).
- 60 • “Compounding Details”: details regarding how the interest should be compounded, including calculation frequency and rate.

5

In the present embodiment of this invention, the Floor element has the following XML definition:

10

```
<!-- Floor -->
5   <!ENTITY % capFloorSpec "premium details
      , strikeRate
      , volatilitySpread
      , discountCurve?
      , forecastCurve?">
10.  <!ELEMENT floor (tradeDate, settlementDate?, startDate, endDate, externalID?,
      %genericSpecDetails: , %floatRateDetails: , %capFloorSpec; , events? )>
<!ATTLIST floor buyer IDREF #REQUIRED writer IDREF #REQUIRED>
20.  <!ELEMENT premiumDetails ( (premiumPercentage | premiumAmount) , premiumDate)>
<!ELEMENT premiumAmount (%currencyAmount; )>
15.  <!ATTLIST premiumAmount %payReceiverAmount;>
<!ELEMENT premiumPercentage (#PCDATA )*>
25.  <!ATTLIST premiumPercentage %payReceiverAmount;>
<!ELEMENT volatilitySpread (#PCDATA )>
<!ELEMENT discountCurve (#PCDATA )>
20.  <!ELEMENT forecastCurvc (#PCDATA )>
```

30

(7) Fixed Rate Loan/Deposit

35

A Fixed Rate Loan/Deposit transaction is one in which one party borrows a sum of
 25 money from another party at a fixed interest rate. For example, a Member borrows from a
 Provider 1 million U.S. Dollars at a fixed interest rate for one year.

40

The Fixed Loan/Deposit element represents such a transaction and includes the
 following sub-elements and attributes:

- “Trade Date”: the date on which the loan has been agreed to by the parties.

30

- “Start Date”: the date on which the loan will begin.
- “End Date”: the date on which the loan will end.

50

55

- 5 • “Lender”: the lender of the loan; this is a reference to a Counterparty element.
- 10 5 • “Borrower”: the borrower of the loan; this is a reference to a Counterparty element.
- 15 10 • “Notional Amount”: the loan amount.
- 20 15 • “Fixed Interest Rate”: the fixed interest rate.
- 25 20 • “Day Count”: the day-count method to be used for calculating interest.
- 30 25 • “Payment Frequency”: the frequency of interest/principal payment.
- 35 30 • “Roll Date”: the specific day each month to be used for payment/settlement of interest/principal.
- 40 35 • “Payment Calendar”: the calendar to be used to generate payment dates.
- 45 40 • “Date Stub”: an indicator for an irregular schedule of loan payments.
- 50 45 • “Anchor Date”: the date to which the payment schedule is anchored, i.e., the end date of the first interest period or specific date of first payment; could be the start of the last interest period if dates generated in reverse.
- 55 50 • “Amortization Details”: details regarding how the loan payment cashflow should be amortized, including amortization method (e.g., single payment at end, equal payments over term of loan).
- “Compounding Details”: details regarding how the loan interest should be compounded, including calculation frequency and rate.

5 In the present embodiment of this invention, the Fixed Loan/Deposit element has the
following XML definition:

```

<!-- Loan and Deposit -->
<!ELEMENT fixedLoan (tradeDate, startDate, endDate, externalId?, %genericSpecDetails;
      , %fixedRateDetails; , events? )>
<!ATTLIST fixedLoan lender IDREF #REQUIRED
      borrower IDREF #REQUIRED>
<!ELEMENT fixedDeposit (tradeDate, startDate, endDate, externalId?, %genericSpecDetails;
      , %fixedRateDetails; , events? )>
<!ATTLIST fixedDeposit lender IDREF #REQUIRED
      borrower IDREF #REQUIRED>
<!ENTITY % genericSpecDetails "notionalAmount
      , dayCount
      , paymentFrequency
      , rollDate
      , anchorDate?
      , paymentCalendar
      , dateStub
      , amortizationDetails?
      , compoundingDetails?">
<!ENTITY % fixedRateDetails " (fixedInterestRate
      | fxRate )">

```

35 25 (8) Floating Rate Loan/Deposit
A Floating Rate Loan/Deposit transaction is one in which one party borrows a sum of
40 money from another party at a variable interest rate, generally based on a floating rate index
(e.g., London Interbank Offered Rate or "LIBOR"). For example, a Member borrows from a
30 Provider 1 million U.S. Dollars at a variable interest rate for two years.

45 The Floating Loan/Deposit element represents such a transaction and includes the
following sub-elements and attributes:

- "Trade Date": the date on which the loan has been agreed to by the parties.

- 50 35 • "Start Date": the date on which the loan will begin.

- 5 • “End Date”: the date on which the loan will end.
- 10 5 • “Lender”: the lender of the loan; this is a reference to a Counterparty element.
- 15 10 • “Borrower”: the borrower of the loan; this is a reference to a Counterparty element.
- 20 15 • “Notional Amount”: the loan amount.
- 25 20 • “Floating Interest Rate”: the floating interest rate.
- 30 25 • “First Fixing Rate”: the interest rate to be used for the first interest calculation period.
- 35 30 15 • “Day Count”: the day-count method to be used for calculating interest.
- 40 35 20 • “Payment Frequency”: the frequency of interest/principal payment.
- 45 40 25 • “Roll Date”: the specific day each month to be used for payment/settlement of interest/principal.
- 50 45 30 • “Payment Calendar”: the calendar to be used to generate payment dates.
- 55 50 25 • “Rate Reset Calendar”: the calendar to be used for reference to business holidays for interest rate resets.
- 60 55 30 • “Date Stub”: an indicator for an irregular schedule of loan payments.
- 65 60 30 • “Anchor Date”: the date to which the payment schedule is anchored, i.e., the end date of the first interest period or specific date of first payment; could be the start of the last interest period if dates generated in reverse.

- 5 • “Amortization Details”: details regarding how the loan payment cashflow should
be amortized, including amortization method (e.g., single payment at end, equal
payments over term of loan).
- 10 5 • “Compounding Details”: details regarding how the loan interest should be
compounded, including calculation frequency and rate.

15 In the present embodiment of this invention, the Floating Loan/Deposit element has
10 the following XML definition:

```
<!-- Loan and Deposit -->
<!ELEMENT floatLoan (tradeDate, startDate, endDate, externalId?, %genericSpecDetails; ,
                     %floatRateDetails; , events? )>
<!ATTLIST floatLoan lender IDREF #REQUIRED
           borrower IDREF #REQUIRED>
<!ELEMENT floatDeposit (tradeDate, startDate, endDate, externalId?, %genericSpecDetails;
                     , %floatRateDetails; , events? )>
<!ATTLIST floatDeposit lender IDREF #REQUIRED
           borrower IDREF #REQUIRED>
<!ENTITY % genericSpecDetails "notionalAmount
                           , dayCount
                           , paymentFrequency
                           , rollDate
                           , anchorDate?
                           , paymentCalendar
                           , dateStub
                           , amortizationDetails?
                           , compoundingDetails?">
<!ENTITY % floatRateDetails "floatingInterestRate
                           , firstFixingRate?
                           , rateResetCalendar">
```

45

50

(9) Foreign Exchange Option

A Foreign Exchange Option (“FX Option”) transaction is one in which one party, in exchange for a premium payment, acquires from another party the right, but not the obligation, to buy (*i.e.*, exercise a put option) or sell (*i.e.*, exercise a call option) a specified quantity of one currency at a specified price on a specified exercise date or during a specified exercise period. For example, a Member pays a premium to a Provider for the right to exercise an option to purchase 1 million Euro for a set price in U.S. Dollars in three months.

The FX Option element represents such a transaction and includes the following sub-elements and attributes:

- 10

 - “Settlement Date”: the date on which the trade will be settled.

15

 - “Premium Details”: the details of the premium to be paid, as either a percentage (“Premium Percentage”) or a specified amount (“Premium Amount”), and the payment date (“Premium Date”).

20

 - “Expiration Date”: the expiration date by which the option must be exercised.
 - “Dealt Amount”: the specified amount of currency to be converted into the currency to be bought or sold upon exercise of the option.

25

 - “Settled Amount”: the amount of currency to be bought or sold upon exercise of the option.
 - “Delivery Date”: the date on which either the cash difference or the underlying contract nominal amount must be exchanged upon exercise of the option.
 - “Delivery Mode”: indicator of whether the cash difference (“Cash”) or the underlying contract nominal amount (“Physical”) must be exchanged upon exercise of the option.

30

 - “Option Type”: the type of option to be exercised (“Put” or “Call”).

- 5 • “Volatility”: the definition of the volatility surface used to calculate the option premium.

- 10 5 • “Buyer”: the buyer of the option to be exercised; this is a reference to a Counterparty element.

- 15 • “Writer”: the recipient of the premium for the option to be exercised; this is a reference to a Counterparty element.

20 In the present embodiment of this invention, the FX Option element has the following XML definition:

```
<!-- FX Option -->
<!ENTITY % fxOptionSpec "tradeDate
                           , settlementDate
                           , externalId?
                           , premiumDetails
                           , expirationDate
                           , deliveryDate
                           , optionType
                           , dealtAmount
                           , strikeRate?
                           , settledAmount
                           , deliveryMode
                           , volatility?">
<!ELEMENT fxOption (%fxOptionSpec; )>
<!ATTLIST fxOption buyer IDREF #REQUIRED writer IDREF #REQUIRED>
<!ELEMENT optionTypec (call | put )>
<!ELEMENT deliveryMode (physical | cash )>
<!ELEMENT volatility (#PCDATA )>
<!ELEMENT call (#PCDATA )>
<!ELEMENT put (#PCDATA )>
<!ELEMENT physical EMPTY>
<!ELEMENT cash EMPTY>
```

35

50

(10) Foreign Exchange Swap

A Foreign Exchange Swap ("FX Swap") transaction is one in which two parties exchange periodic payment streams, each in a different currency. The first payment stream is delivered at the beginning of the transaction period and the second payment is delivered at the end of the transaction period. The payments may be based upon a specified interest rate. For example, a Member buys a payment stream of 3 million Euro from a Provider in exchange for a payment stream of 1 million U.S. Dollars to be paid six months after the first payment stream.

The FX Swap element represents such a transaction and includes the following sub-elements and attributes:

- “Trade Date”: the date on which the trade has been agreed to by the parties.
 - “Near Leg Value Date”: the date on which the final payment of the first leg (the “Near Leg”) of the swap will be paid.
 - “Far Leg Value Date”: the date on which the final payment of the second leg (the “Far Leg”) of the swap will be paid.
 - “Notional Amount”: the amount used as the basis for calculating the payment streams to be exchanged.
 - “Near Leg Settled Amount”: the amount that will be paid under the Near Leg; alternative to Near Leg FXRate.
 - “Near Leg FXRate”: the foreign exchange rate of the Near Leg; alternative to Near Leg Settled Amount.
 - “Far Leg Settled Amount”: the amount that will be paid under the Far Leg; alternative to Far Leg FXRate.
 - “Far Leg FXRate”: the foreign exchange rate of the Far Leg; alternative to Far Leg Settled Amount.

5

In the present embodiment of this invention, the FX Swap element has the following XML definition:

```

10      5 <!-- FX Swap -->
      5 <!ENTITY % fxSwapSpec "tradeDate
          , externalId?
          , nearLegValueDate
          , farLegValueDate
          , notionalAmount
          , (nearLegFXRate | nearLegSettledAmount )
          , (farLegFXRate | farLegSettledAmount )">
15      10 <!ELEMENT fxSwap (%fxSwapSpec; )>
20      20 <!ELEMENT nearLegValueDate (#PCDATA )>
          <!ELEMENT farLegValueDate (#PCDATA )>
15      15 <!ELEMENT nearLegFXRate (fxRate )>
          <!ELEMENT farLegFXRate (fxRate )>
25      25 <!ELEMENT nearLegSettledAmount (%currencyAmount; )>
          <!ATTLIST nearLegSettledAmount %payReceiver; >
          <!ELEMENT farLegSettledAmount (%currencyAmount; )>
20      20 <!ATTLIST farLegSettledAmount %payReceiver; >

```

30

(11) Cross-Currency Fixed-Fixed Swap

A Cross-Currency Fixed-Fixed Swap is a type of interest rate swap in which two parties exchange periodic payment streams based on fixed interest rates each in a different currency.

The Cross-Currency Fixed-Fixed Swap element represents such a transaction and includes the following sub-elements and attributes:

- “Trade Date”: the date on which the trade has been agreed to by the parties.

30

- “Start Date”: the date on which the exchanged payments will begin.

45

- “End Date”: the date on which the exchanged payments will end.

50

- 5 • “Notional Amount”: the amount used as the basis for calculating the payment streams to be exchanged.
- 10 5 • “Fixed Leg Details”: the details of the fixed interest payments; separate information for each of the two fixed legs.
- 15 10 • “Events”: the various payment and calculation events in the swap transaction, including cash payment, principal payment, interest payment, interest calculation, compound interest calculation, and interest rate reset information.

20 In the present embodiment of this invention, the Cross-Currency Fixed-Fixed Swap element has the following XML definition:

```
<!-- Cross Currency Fixed Fixed Swap -->
<!ELEMENT crossCurrencyFixedSwap (%tenor.elements: , fixedLegDetails , fixedLegDetails , events?
15 )>
<!ATTLIST crossCurrencyFixedSwap notionalAmount (Yes|No) #REQUIRED >
```

30 **(12) Cross-Currency Float-Float Swap**

20 A Cross-Currency Float-Float Swap is a type of interest rate swap in which two parties exchange periodic payment streams based on a floating rate index (e.g., LIBOR), each in a different currency.

35 The Cross-Currency Float-Float Swap element represents such a transaction and includes the following sub-elements and attributes:

- 25 • “Trade Date”: the date on which the trade has been agreed to by the parties.
- 40 • “Start Date”: the date on which the exchanged payments will begin.
- 45 • “End Date”: the date on which the exchanged payments will end.
- 50 30 • “Notional Amount”: the amount used as the basis for calculating the payment streams to be exchanged.

5

- “Float Leg Details”: the details of the floating interest payments; separate information for each of the two fixed legs.

10

- 5 • “Events”: the various payment and calculation events in the swap transaction, including cash payment, principal payment, interest payment, interest calculation, compound interest calculation, and interest rate reset information.

15

In the present embodiment of this invention, the Cross-Currency Float-Float Swap element has the following XML definition:

```
<!-- Cross Currency Float Float Swap -->
<!ELEMENT crossCurrencyFloatFloatSwap (%tenor.elements; , floatLegDetails , floatLegDetails , events?)>
<!ATTLIST crossCurrencyFloatFloatSwap notionalAmount (Yes | No) #REQUIRED >
```

25

(13) Cross-Currency Fixed-Float Swap

A Cross-Currency Fixed-Float Swap is a type of interest rate swap in which two parties exchange periodic payment streams, where one payment stream is based on a fixed interest rate and the other payment stream is based on a floating rate index (e.g., LIBOR), 30 each in a different currency.

The Cross-Currency Fixed-Float Swap element represents such a transaction and includes the following sub-elements and attributes:

- “Trade Date”: the date on which the trade has been agreed to by the parties.
- 25 • “Start Date”: the date on which the exchanged payments will begin.
- 40 • “End Date”: the date on which the exchanged payments will end.
- 45 • “Notional Amount”: the amount used as the basis for calculating the payment streams to be exchanged.
- 30 • “Fixed Leg Details”: the details of the fixed interest payments for the fixed leg.

50

- 5 • “Float Leg Details”: the details of the floating interest payments for the floating leg.
- 10 5 • “Events”: the various payment and calculation events in the swap transaction, including cash payment, principal payment, interest payment, interest calculation, compound interest calculation, and interest rate reset information.

15 In the present embodiment of this invention, the Cross-Currency Fixed-Float Swap
20 10 element has the following XML definition:

```
<!-- Cross Currency Fixed Float Swap -->
<!ELEMENT crossCurrencyFixedFloatSwap (%tenor.elements; , fixedLegDetails, floatLegDetails, events?)>
<!ATTLIST crossCurrencyFixedFloatSwap notionalAmount (Yes | No ) #REQUIRED >
```

25 15

30 A Forward Rate Agreement transaction is one in which one party buys a single
35 floating rate payment in exchange for a single fixed rate payment. The fixed rate payment
40 amount is determined by applying a fixed rate of interest to the notional amount of the
45 transaction, while the floating rate payment amount is determined by sampling the value of a
50 specified floating rate option on a specified date and applying the sampled rate to the notional
55 amount. The parties settle the Forward Rate Agreement by netting the effects of the two
60 payments into a single payment made by one or the other of the parties: if the floating rate
65 amount due is greater than the fixed rate amount due, then the floating rate payer pays the
70 excess to the fixed rate payer; conversely, if the fixed rate amount due is greater than the
75 floating rate amount due, then the fixed rate payer pays the excess to the floating rate payer.
80 Settlement occurs at the beginning of the transaction subject to future discounting (*i.e.*,
85 payment of difference in fixed and floating rates).

30 The Forward Rate Agreement element represents such a transaction and includes the
35 following sub-elements and attributes:

- 50 • “Trade Date”: the date on which the trade has been agreed to by the parties.

5

- “Settlement Date”: the date on which payment settlement will be completed.

10

- “Start Date”: the date on which the transaction will begin.

5

- “End Date”: the date on which the transaction will end.

15

- “Adjusted Start Date”: the date on which the transaction will begin, adjusted for holidays.

10

- “Adjusted End Date”: the date on which the transaction will end, adjusted for holidays.

20

- “Notional Amount”: the amount used as the basis for calculating the payments to be exchanged.

25

- “Fixed Interest Rate”: the fixed interest rate for the fixed rate payment.

30

- “Interest Index”: the details of the floating interest index to be used for the floating rate payment.

20

- “Day Count”: the day-count method to be used for calculating interest.

35

- “Payment Frequency”: the frequency of interest/principal payment.

40

- “Roll Date”: the specific day each month to be used for payment/settlement of interest/principal.

25

- “Roll Convention”: the convention to be used for rolling the payment dates in the event the date falls on a holiday.

45

30

- “Holiday Calendar”: the calendar to be used for reference to business holidays.

50

55

- 5 • “Fixing Date”: the date on which the rate to be used for settlement is fixed.
- 10 • “Rate Reset Calendar”: the calendar to be used for determining the dates on which
5 to reset floating interest rates.
- 15 • “Buyer”: the buyer of the floating rate payment; this is a reference to a
Counterparty element.
- 20 • “Seller”: the seller of the floating rate payment; this is a reference to a
Counterparty element.
- 25 • “Premium Details”: the details of the premium to be paid, as either a percentage
15 (“Premium Percentage”) or a specified amount (“Premium Amount”), and the
payment date (“Premium Date”).

In the present embodiment of this invention, the Forward Rate Agreement element
30 has the following XML definition:

```
<!-- Forward Rate Agreement -->
<!ELEMENT forwardRateAgreement (tradeDate, settlementDate?, startDate, endDate,
    externalId?, adjustedStartDate, adjustedEndDate ,
    notionalAmount, dayCount, rollConvention, rollDate,
    holidayCalendar, fixedInterestRate, interestIndex,
    fixingDate, rateResetCalendar, premiumDetails? )>
<!ATTLIST forwardRateAgreement buyer IDREF #REQUIRED>
<!ATTLIST forwardRateAgreement seller IDREF #REQUIRED>
<!ELEMENT adjustedStartDate (#PCDATA )>
<!ELEMENT adjustedEndDate (#PCDATA )>
<!ELEMENT fixingDate (#PCDATA)>
```

45

50

(15) Customized Trade

In addition to the financial transactions represented by the elements described above, the present embodiment of this invention supports customized trades and transactions created by Members and/or Providers, so long as such transactions are permitted by applicable law.

5 Such customized transactions might include hybrid trades, where one or more aspects of one type of trade are combined with those of another. For example, a party might structure a foreign exchange "swaption" in which a stream of periodic payments in one currency is exchanged for the right to buy a specified quantity of another currency at a specified price on a specified date.

10 FinXML enables the representation of customized transactions through the combination of elements that comprise different types of transactions. Using FinXML, a party can specify the element fields and values that it wishes to comprise the particular customized transactions. The Customized Trade element represents such a transaction and includes the following sub-elements and attributes:

- “Field Name”: a particular component included in the transaction; separate information for each component; paired with “Field Value”.
 - “Field Value”: the value of a particular component included in the transaction; separate information for each component; paired with “Field Name”.

20

 - “Buyer”: the buyer of the customized trade; this is a reference to a Counterparty element.
 - “Seller”: the seller of the customized trade; this is a reference to a Counterparty element.

25

In the present embodiment of this invention, the Customized Trade element has the following XML definition:

```
5      <!-- Customized Trade -->
<!ELEMENT customizedTrade ( (fieldName , fieldValue ) * )>
<!ATTLIST customizedTrade buyer IDREF #REQUIRED>
<!ATTLIST customizedTrade seller IDREF #REQUIRED>
10     <!ELEMENT fieldName (#PCDATA )>
<!ELEMENT fieldValue (#PCDATA )>
```

15

10 (c) **Trade Specific Elements**

In the present embodiment of this invention, FinXML includes a number of elements
that represent details common to one or more of the Trade Type elements 530. Such
elements may also be included in customized trades.

15 (1) **Generic Trade Details**

25 Generic trade details include information relating to notional amounts and interest
rate, amortization, and compounding calculations that are common to different types of
trades. The “Generic Spec Details” element represents such information and includes the
30 following sub-elements and attributes:

- 20 • “Notional Amount”: the transaction amount.
- 35 • “Day Count”: the day-count method to be used for calculating interest.
- 25 • “Payment Frequency”: the frequency of interest/principal payment (e.g., monthly,
quarterly, semi-annually).
- 40 • “Roll Date”: the specific day each month to be used for payment/settlement of
interest/principal.
- 45 30 • “Anchor Date”: the date to which the payment schedule is anchored, i.e., the end
date of the first interest period or specific date of first payment.

50

55

- 5 • “Payment Calendar”: the calendar to be used for reference to business holidays.
- 10 • “Date Stub”: an indicator for a schedule of loan payments in which the payment period differs (*i.e.*, is offset from the start of) from all other payment periods.
- 15 • “Amortization Details”: details regarding how the loan payment cashflow should be amortized, including amortization method (*e.g.*, single payment at end, equal payments over term of loan).
- 20 • “Compounding Details”: details regarding how the loan interest should be compounded, including calculation frequency and rate.

In the present embodiment of this invention, the Generic Spec Details element has the following XML definition:

```
25                    15 <!ENTITY % genericSpecDetails "notionalAmount  
30                    , dayCount  
                     , paymentFrequency  
                     , rollDate  
                     , anchorDate?  
                     , paymentCalendar  
                     , dateStub  
                     , amortizationDetails?  
                     , compoundingDetails?">
```

25

(2) Fixed Rate Details

40 Fixed rate details include information relating to fixed interest rates. The “Fixed Spec Details” element represents such information and includes the following sub-elements and attributes:

- 45 30 • “Fixed Interest Rate”: the fixed interest rate.
- “FX Rate”: the foreign exchange rate at which a trade will be executed.

50

55

5 In the present embodiment of this invention, the Fixed Spec Details element has the
following XML definition:

```
<!ENTITY % fixedSpecDetails "fixedInterestRate | fxRate">
```

10

5

(3) Floating Rate Details

Floating rate details include information relating to floating interest rates that are
15 based on a floating rate index (e.g., LIBOR). The “Floating Spec Details” element represents
such information and includes the following sub-elements and attributes:

- 10 • “Floating Interest Rate”: the floating interest rate.
- 20 • “First Fixing Rate”: the interest rate to be used for the first interest calculation
period.
- 25 15 • “Rate Reset Calendar”: the calendar to be used for reference to business holidays
for interest rate resets.

30 In the present embodiment of this invention, the Floating Spec Details element has the
following XML definition:

```
<!ENTITY % floatingSpecDetails "floatingInterestRate, firstFixingRate?, rateResetCalendar">
```

35

(4) Fixed Leg Details

A number of the transactions described above include multiple “legs,” where each leg
40 25 is a series of payments or cashflows. Such legs can be “fixed” or “floating.”

45 A “fixed leg” is a payment stream based on a fixed interest rate. The “Fixed Leg
Details” elements represents information regarding the fixed leg of a trade and includes
generic trade details (described above in “Generic Spec Details” element), fixed rate details
(described above in “Fixed Spec Details” element), financial events details (described below
30 in “Events” element), and the following additional sub-elements and attributes:

- “Leg ID”: identifier of a particular leg of a trade.

50

- 5 • “Payer”: the payer of the fixed leg in a trade; this is a reference to a Counterparty element.
- 10 5 • “Receiver”: the recipient of the proceeds of the fixed leg in a trade; this is a reference to a Counterparty element.

15 In the present embodiment of this invention, the Fixed Leg Details element has the following XML definition:

20 10 <!ELEMENT fixedLegDetails (%genericSpecDetails; , %fixedRateDetails; , events?)>
 <!ATTLIST fixedLegDetails legID ID #REQUIRED >
 <!ATTLIST fixedLegDetails payer IDREF #REQUIRED >
 <!ATTLIST fixedLegDetails receiver IDREF #REQUIRED >

15 25 **(5) Floating Leg Details**

20 A “floating leg” is a payment stream based on a floating interest rate. The “Float Leg Details” elements represents information regarding the floating leg of a trade and includes generic trade details (described above in “Generic Spec Details” element), floating rate details (described above in “Float Spec Details” element), financial event details (described below in “Events” element), and the following additional sub-elements and attributes:

- 25 35 • “Leg ID”: identifier of a particular leg of a trade.
 • “Payer”: the payer of the floating leg in a trade; this is a reference to a Counterparty element.
- 40 45 • “Receiver”: the recipient of the proceeds of the floating leg in a trade; this is a reference to a Counterparty element.

45 In the present embodiment of this invention, the Float Leg Details element has the following XML definition:

```
<!ELEMENT floatLegDetails (%genericSpecDetails: . %floatRateDetails: . events? )>
<!ATTLIST floatLegDetails legID ID #REQUIRED >
<!ATTLIST floatLegDetails payer IDREF #REQUIRED >
<!ATTLIST floatLegDetails receiver IDREF #REQUIRED >
```

(d) Financial Event Elements

In the present embodiment of this invention, FinXML includes a number of elements

10 that represent details common to certain Trade Type elements 530, including customized trades, that relate to optional events during the life cycle of a trade such as premium payment, interest payment, contingent payment, and interest calculation. “Events” element 900, shown in FIG. 6, describes such information and includes the following sub-elements: “Cash Payment” 910, “Principal Payment” 920, “Interest Payment” 930, “Interest Calculation” 940, 15 “Compound Interest Calculation” 950, and “Contingent Payment” 960.

In the present embodiment of this invention, Events element 900 has the following XML definition:

```
<!ELEMENT evnts ((cashPayment | principalPayment | interestPayment | contingentPayment |  
interestCalculation | compoundInterestCalculation )+ >  
<!ATTLIST events id ID #IMPLIED >
```

(1) Cash Payment

Cash Payment element 910 describes information relating to cash payments to be made as a part of certain trades, and includes the following sub-elements and attributes:

- “Currency”: the currency of the cash payment.
 - “Amount”: the amount of the cash payment.
 - “Payment Date”: the date on which the cash payment is to be made.
 - “ID”: the identifier of the particular cash payment.
 - “Type”: the indicator of type of payment (e.g., “Premium” or “Fees”).

- 5 • “Payer”: the payer of the cash payment; this is a reference to a Counterparty element.
- 10 5 • “Receiver”: the recipient of the cash payment; this is a reference to a Counterparty element.

15 In the present embodiment of this invention, Cash Payment element 910 has the following XML definition:

20 10 <!ELEMENT cashPayment (currency, amount, paymentDate)>
 <!ATTLIST cashPayment id ID #REQUIRED
 type (Premium | Fees) #REQUIRED
 payer IDREF #REQUIRED
 receiver IDREF #REQUIRED >

15 25 **(2) Principal Payment**

20 Principal Payment element 920 describes information relating to principal payments to be made as a part of certain trades, and includes the following sub-elements and attributes:

- 30 20 • “Currency”: the currency of the principal payment.
- 35 25 • “Amount”: the amount of the principal payment.
- “Payment Date”: the date on which the principal payment is to be made.
- “ID”: the identifier of the particular principal payment.
- “Payer”: the payer of the principal payment; this is a reference to a Counterparty element.
- 40 30 • “Receiver”: the recipient of the principal payment; this is a reference to a Counterparty element.

45 In the present embodiment of this invention, Principal Payment element 920 has the following XML definition:

5 **<!ELEMENT principalPayment (currency, amount, paymentDate)>**
 <!ATTLIST principalPayment id ID #REQUIRED
 payer IDREF #REQUIRED
 receiver IDREF #REQUIRED >

10 5

(3) Interest Payment

15 Interest Payment element 930 describes information relating to interest payments to
be made as a part of certain trades, and includes the following sub-elements and attributes:

- 10 • “Currency”: the currency of the interest payment.
- “Amount”: the amount of the interest payment.
- 20 • “Payment Date”: the date on which the interest payment is to be made.
- “Start Date”: the start date of the interest period to which the interest payment
pertains.
- 15 • “End Date”: the end date of the interest period to which the interest payment
pertains.
- 30 • “ID”: the identifier of the particular interest payment.
- 20 • “Payer”: the payer of the interest payment; this is a reference to a Counterparty
element.
- 35 • “Receiver”: the recipient of the interest payment; this is a reference to a
Counterparty element.
- 25 • “Interest Type”: the indicator of type of interest payment (e.g., “Coupon”,
“Swap”, “Loan”, “Deposit”, or “Other”).
- 40 • “Calculations”: the identifier of the particular interest calculation periods.

45 In the present embodiment of this invention, Interest Payment element 930 has the
following XML definition:

50

```
5 <ELEMENT interestPayment (currency, paymentDate, startDate, endDate) >
<!ATTLIST interestPayment id ID #REQUIRED
      payer IDREF #REQUIRED
      receiver IDREF #REQUIRED
      10    interestType (Coupon | Swap | Loan | Deposit | Other) #IMPLIED
            calculations IDREFS #REQUIRED >
```

(4) Contingent Payment

Contingent Payment element 960 describes information relating to contingent payments to be made in the settlement of certain trades after the exercise of an option, and includes the following sub-elements and attributes:

- “Underlying Amount”: the amount of the option-underlying instrument.
 - “Settlement Amount”: the amount to be paid in settlement of the exercise of the option in return for the underlying instrument.
 - “Expiration Date”: the date of expiry of the option.
 - “Exercise Begin Date”: the first date on which the option may be exercised.
 - “Exercise End Date”: the last date on which the option may be exercised.
 - “Exercise Rule”: the rule governing normal exercise of the option (e.g., “American” – the option may be exercised on any day within a given period; “European” – the option may only be exercised on the option expiration date).
 - “Exercise Condition”: any conditions that must be met to permit exercise of the option (e.g., the 3-month LIBOR rate must be greater than 4.5% on the exercise date).
 - “Volatility”: the volatility value to be used when valuing the option.

- 5 • “ID”: the identifier of the particular interest payment.
- 10 • “Payer”: the party responsible for delivering the option underlying instrument; this party will receive the settlement amount in exchange for the option underlying instrument.
- 15 • “Receiver”: the recipient of the option-underlying instrument; this party will pay the settlement amount as the price for exercising the option.
- 20 • “Option Type”: the nature of the option (e.g., “Call” – an option to buy the underlying instrument at the exercise price; “Put” – an option to sell the underlying instrument at the exercise price).
- 25 • “Delivery Type”: an indicator describing whether the Payer will physically deliver the option underlying instrument to the Receiver or, alternatively, that the transaction will be settled for cash where the option writer will, upon exercise, pay to the option holder the difference between the value of the underlying instrument and the exercise price.

30 In the present embodiment of this invention, Contingent Payment element 960 has the
20 following XML definition:

```
<!ELEMENT contingentPayment (underlyingAmount, settlementAmount, expirationDate, exerciseBeginDate,  
exerciseEndDate, exerciseRule, exerciseCondition, volatility)>  
<!ATTLIST contingentPayment id ID #REQUIRED  
      payer IDREF #REQUIRED  
      receiver IDREF #REQUIRED  
      optionType (call | put)#REQUIRED  
      deliveryType (deliverable | non-deliverable) #REQUIRED>  
<!ELEMENT underlyingAmount (currency, amount)>  
<!ELEMENT settlementAmount (currency, amount)>  
30            <!ELEMENT exerciseBeginDate (#PCDATA)>  
          <!ELEMENT exerciseEndDate (#PCDATA)>  
          <!ELEMENT exerciseRule (#PCDATA)>  
          <!ELEMENT exerciseCondition (#PCDATA)>  
          <!ELEMENT volatility (#PCDATA)>
```

5

(5) Interest Calculation

Interest Calculation element 940 describes information relating to an interest amount

10 calculated for a given period within a particular interest payment, and includes the following
5 sub-elements and attributes:

- “ID”: the identifier of the particular interest calculation period.
- “Resets”: the identifier of the particular rate reset series.
- “Notional Amount”: the amount involved in the interest calculation.
- “Calculation Date”: the date on which the interest calculation is performed.
- “Start Date”: the start date of the interest period for which the interest calculation
10 is to be performed.
- “End Date”: the end date of the interest period for which the interest calculation is
15 to be performed.
- “Amount”: the calculated interest amount.
- “Day Count”: the day-count method to be used for performing the interest
20 calculation.
- “%InterestRate.Elements”: definition of the type of interest rate involved (e.g.,
25 “Fixed” or “Floating”).

30 In the present embodiment of this invention, Interest Calculation element 940 has the
25 following XML definition:
40

```
<!ELEMENT interestCalculation ((%interestRate.elements; )?, notionalAmount, calculationDate, startDate,  
45       endDate, amount?, dayCount )>  
<!ATTLIST interestCalculation id ID #REQUIRED  
30        resets IDREFS #IMPLIED >
```

50

(6) Compound Interest Calculation

Compound Interest Calculation element 950 describes information relating to a compound interest amount calculated for a given period within a particular interest payment, and includes the following sub-elements and attributes:

- “ID”: the identifier of the particular interest calculation period.
 - “Rate”: the identifier of the particular interest rate.
 - “Resets”: the identifier of the particular rate reset series.
 - “Notional Amount”: the amount involved in the compound interest calculation.
 - “Calculation Date”: the date the compound interest calculation is performed.
 - “Start Date”: the start date of the interest period for which the compound interest calculation is to be performed.
 - “End Date”: the end date of the interest period for which the compound interest calculation is to be performed.
 - “Amount”: the calculated compound interest amount.
 - “%InterestRate.Elements”: definition of the type of interest rate involved (e.g., “Fixed” or “Floating”).

In the present embodiment of this invention, Compound Interest Calculation element has the following XML definition:

5

```
<!ELEMENT compoundInterestCalculation ((fixedInterestRate | floatingInterestRate)?.
      calculationDate, startDate, endDate, amount)>
<!ATTLIST compoundInterestCalculation id ID #REQUIRED
      resets IDREF #REQUIRED
      rate IDREF #IMPLIED>
```

10

15

(e) Calculation Elements

10 In the present embodiment of this invention, FinXML includes a number of elements
that represent details regarding calculations to be performed in certain Trade Type elements
530, including customized trades. These elements relate to compounding, amortization, and
calculation frequency.

20

15 (1) Compounding Details

25

The “Compounding Details” element describes information relating to any
compounding calculations that need to be performed in a particular transaction. This
typically arises when the actual interest payment frequency is less than the interest calculation
frequency. For example, if interest is calculated every three months but paid every 6 months,
30 then the interest calculated at the end of the 3-month period would be compounded and paid
along with the interest calculated for the fourth through sixth months. The Compounding
Details element includes the following sub-element:

35

- “Calculation Frequency”: the frequency at which interest calculations should be
performed in a multi-period transaction.

25

40

In the present embodiment of this invention, the Compounding Details element has
the following XML definition:

```
<!ELEMENT compoundingDetails (calculationFrequency)>
```

45

30

50

(2) Amortization Details

The “Amortization Details” element describes information relating to any amortization calculations that need to be performed in a particular swap transaction. If the amortization method is defined to be “bullet”, principal will be paid in one lump sum at maturity, whereas under “equal” amortization, principal will be paid in equal installments during the life of the swap transaction. The Amortization Details element includes the following sub-elements and attributes:

- “Amortization Frequency”: the frequency at which amortization will be performed in a particular transaction (e.g., semi-annual or annual).
 - “Amortization Method”: the amortization method (e.g., “bullet” or “equal”).

In the present embodiment of this invention, the Amortization Details element has the following XML definition:

```
15  <!ELEMENT amortizationDetails (amortizationFrequency )>
    <!ATTLIST amortizationDetails amortizationMethod %amortMethod; #REQUIRED>
```

(3) Calculation Frequency

The “Calculation Frequency” element describes information relating to the frequency of a particular calculation to be performed. The Calculation Frequency element includes the following sub-elements and attributes:

- “Convention”: the particular calculation methodology based on the market convention (e.g., “IMM”, “FRN”, “Eurodollar”, or “Normal”).
 - “End of Month”: indicator of whether the particular calculation should be to the end of the month.
 - “Term”: the period of time for a single calculation period (e.g., 3-months, months, etc.).

5

In the present embodiment of this invention, the Calculation Frequency element has the following XML definition:

10

```
<!ELEMENT calculationFrequency (term )>
<!ATTLIST calculationFrequency convention (IMM | FRN | Eurodollar | Normal ) 'Normal'
      endOfMonth (Yes | No ) #REQUIRED >
```

15

(4) Payment Frequency

20

The "Payment Frequency" element describes information relating to the frequency of a particular payment to be made. The Payment Frequency element includes the following sub-elements and attributes:

- "Convention": the particular calculation methodology based on the market convention (e.g., "IMM", "FRN", "Eurodollar", or "Normal").

25

- "End of Month": indicator of whether the particular payment should be moved to the end of the month.

30

- "Term": the term of the interest index used in calculating the particular payment (e.g., 3-months, 6-months, etc.).

35

In the present embodiment of this invention, the Payment Frequency element has the following XML definition:

40

```
<!ELEMENT paymentFrequency (term )>
<!ATTLIST paymentFrequency convention (IMM | FRN | Eurodollar | Normal ) 'Normal'
      endOfMonth (Yes | No ) #REQUIRED >
```

45

(5) Amortization Frequency

The "Amortization Frequency" element describes information relating to the frequency of a particular amortization to be performed. The Amortization Frequency element includes the following sub-elements and attributes:

50

- 5 • “Convention”: the particular calculation methodology based on the market convention (e.g., “IMM”, “FRN”, “Eurodollar”, or “Normal”).
- 10 5 • “End of Month”: indicator of whether the particular amortization should be moved to the end of the month.
- 15 10 • “Term”: the period of time for a single amortization calculation period (e.g., 3-months, 6-months, etc.).

20 15 In the present embodiment of this invention, the Payment Frequency element has the following XML definition:

```
<!ELEMENT paymentFrequency (term )>
<!ATTLIST paymentFrequency convention (IMM | FRN | Eurodollar | Normal ) 'Normal'
      endOfMonth (Yes | No ) #REQUIRED >
```

25

ii. Reference Data

30 Reference data describes the profile information specific to Members and Providers that will be referenced in any transactions engaged in by such parties. The FinXML syntax
20 represents this profile information with the following elements: “Organization” element 710 (FIG. 4), “Contact Information” element 730 (FIG. 4), “Address” element 765 (FIG. 4),
35 “Credit Rating” element 805 (FIG. 4), “Legal Entity” element 605 (FIG. 5), and “Book” element 625 (FIG. 5).

25 (a) Organization

40 Organization element 710 (as shown in FIG. 4) describes the organizational information regarding a Disclosed Party 705. Organization element 710 includes the following sub-elements and attributes:

- 45 • “Organization Name” 715: the full name of the organization.
- 30 • “Organization Short Name” 720: the short name of the organization.
- 50 • “Address” 725: the address of the organization.

5 In the present embodiment of this invention, Organization element 710 has the
following XML definition:

```
<!ELEMENT organization (organizationShortName, organizationName, address )>
<!ELEMENT organizationShortName (#PCDATA )>
<!ELEMENT organizationName (#PCDATA )>
```

10

15 (b) Contact Information

10 Contact Information element 730 (as shown in FIG. 4) describes the information
necessary to contact a Disclosed Party 705 during the transaction process. Contact
Information element 730 includes the following sub-elements and attributes:

- 20
- “Contact Name” 735: name of the specific contact within the party.
 - “Contact ID”: the identifier of the particular contact.
 - “Telephone” 740: the telephone details of the party.
 - 15 • “Fax” 745: the fax details of the party.
 - “Telex” 750: the telex details of the party.
 - “Email” 755: the electronic mail details of the party.
 - “URL” 760: the Uniform Resource Locator details of the party.

30 In the present embodiment of this invention, Contact Information element 730 has the

20 following XML definition:

```
<!ELEMENT contactInformation (contactName, (telephone | fax | telex | email | url)* )>
<!ATTLIST contactInformation contactID #REQUIRED
      default (Y | N ) #REQUIRED >
<!ELEMENT contactName (#PCDATA )>
<!ELEMENT telex (#PCDATA )>
<!ELEMENT telephone (#PCDATA )>
<!ELEMENT fax (#PCDATA )>
<!ELEMENT email (#PCDATA )>
<!ELEMENT URL (#PCDATA )>
```

30

45

50

5

(c) Address

Address element 765 (as shown in FIG. 4) describes the registered address information of the Disclosed Party 705. Address element 765 includes the following sub-elements and attributes:

- 10 • “Address1” 770: the first line of the street address of the party.
 • “Address2” 775: the second line of the street address of the party.
 • “City” 780: the city of the party.
 • “State-Province-County” 785: the state, province, and/or county of the party.
 • “Zip Postal Code” 790: the zip or postal code of the party.
15 • “Country” 795: the country of the party.
 • “SWIFT Address” 800: the Bank-identifier Code (“BIC”) of the party (as assigned
20 by S.W.I.F.T. sc).

In the present embodiment of this invention, Address element 765 has the following

25 XML definition:

```
<!ELEMENT address (address1, address2, city, stateProvinceCounty, zipPostalCode, country, swiftAddress?)>
<!ELEMENT address1 (#PCDATA )>
<!ELEMENT address2 (#PCDATA )>
<!ELEMENT city (#PCDATA )>
20 <!ELEMENT stateProvinceCounty (#PCDATA )>
<!ELEMENT zipPostalCode (#PCDATA )>
<!ELEMENT country (#PCDATA )>
35 <!ELEMENT swiftAddress (#PCDATA )>
```

25

(d) Credit Rating

Credit Rating element 805 (as shown in FIG. 4) describes the details of the credit rating of the Disclosed Party 705 or Undisclosed Party 835, as rated by standard credit rating agencies. Credit Rating element 805 includes the following sub-elements and attributes:

- 40 30 • “Agency” 810: the name of the credit rating agency that provided the credit rating
 of the party.

50

55

- 5 • “Rating” 815: the actual rating value (e.g., AAA, BB, etc.) of the party provided by the credit rating agency.
- 10 5 • “Country” 820: the country to which the party is assigned for purposes of the credit rating by the credit rating agency.
- 15 10 • “Industry Group” 825: the industry group to which the party is assigned for purposes of the credit rating by the credit rating agency.
- 20 15 • “Industry” 830: the industry to which the party is assigned for purposes of the credit rating by the credit rating agency.

In the present embodiment of this invention, Credit Rating element 805 has the following XML definition:

```
25            15        <!ELEMENT creditRating (agency, rating, country, industryGroup, industry )>
26            16        <!ELEMENT agency (#PCDATA )>
27            17        <!ELEMENT rating (#PCDATA )>
28            18        <!ELEMENT name (#PCDATA )>
29            19        <!ELEMENT industryGroup (#PCDATA )>
30            20        <!ELEMENT industry (#PCDATA )>
```

35 (e) Legal Entity

Legal Entity element 605 (as shown in FIG. 5) describes the details of any legal entities (e.g., subsidiaries or affiliate companies) associate with an Internal Party 600 (as shown in FIG. 5). Legal Entity element 605 includes the following sub-elements and attributes:

- 40 30 • “ID” 608: the identifier of the legal entity.

- 45 30 • “Short Name” 610: the short name of the legal entity.

- 50 30 • “Description” 615: the description of the legal entity.

5

- “Parent” 620: the name of the parent organization of the legal entity.

In the present embodiment of this invention, Legal Entity element 605 has the

10 5 following XML definition:

```
<!ELEMENT legalEntity (shortName, description, parent)>
<!ATTLIST legalEntity id ID #IMPLIED>
```

15

10 (f) Trading Book

Book element 625 (as shown in FIG. 5) describes the details of any internal trading
20 book associated with the transaction by a party. Book element 625 includes the following
sub-elements and attributes:

- “ID”: the identifier of the trading book.
- 15 • “Type”: the type of trading book.
- 20 • “Short Name” 630: the short name of the trading book.
- 30 • “Name” 635: the full name of the trading book.
- 35 • “Description” 640: the description of the trading book.
- 40 • “Reporting Currency” 645: the reporting currency of the trading book.

25 In the present embodiment of this invention, Book element 625 has the following
40 XML definition:

```
<!ELEMENT book (shortName, name, description, reportingCurrency )>
<!ATTLIST book id ID #REQUIRED
    type CDATA #IMPLIED >
```

50

5

iii. Market Data

Market data describes information obtained from market sources for use in financial transactions. FinXML represents this information with the following elements: "Floating Interest Rate" element and "Interest Index" element.

(1) Floating Interest Rate

The "Floating Interest Rate" element describes information relating to the floating interest rate that can be used in a transaction. The Floating Interest Rate element includes the following sub-elements and attributes:

- "ID": the identifier of the particular floating interest rate definition.
- "Interest Index": the details of a particular index used for a floating interest rate, including currency ("Currency"), term ("Term"), and name ("Index Name").
- "Spread": the differential (plus or minus) to be applied to the index rate in order to determine the floating interest rate.

In the present embodiment of this invention, the Floating Interest Rate element has the

following XML definition:

```
<!ELEMENT floatingInterestRate (interestIndex, spread )>
<!ATTLIST floatingInterestRate id ID #IMPLIED >
```

25

(2) Interest Index

The "Interest Index" element describes information relating to the interest index used to calculate the floating interest rate. The Interest Index element includes the following sub-elements and attributes:

- "ID": the identifier of the particular interest index.
- "Currency": the currency of the interest index.

50

55

- 5
- “Term”: the term of the interest index (e.g., 3-months, 6-months, etc.).
 - “Index Name”: the name of the interest index (e.g., “LIBOR”).
- 10

5
In the present embodiment of this invention, the Interest Index element has the following XML definition:

15

```
<!ELEMENT interestIndex (currency, term, indexName)>
<!ATTLIST interestIndex id ID #IMPLIED>
10 <!ELEMENT indexName (#PCDATA)>
```

20

2. “Connect” Processor

In the present embodiment of this invention, the Connect Processor 20 (as shown in

25 FIG. 1) provides the means for communicating information related to financial transactions between users (i.e., Members and Providers) and the CFOWeb System. Connect Processor 20 performs this function by converting FinXML (or other XML) documents to/from financial (Java) objects using proprietary stylesheets created in XSL, known as “FinScript”, as will be described below.

30
20 In the present embodiment of this invention, both Connect Processor 20 and Connect Messaging Server 90 process messages between users and the CFOWeb System and convert FinXML (or other XML) documents to/from financial (Java) objects. Whereas Connect Processor 20 performs such conversion between FinXML (or other XML) documents and the proprietary objects of Members and Providers, Connect Messaging Server 90 performs such 35 conversion between FinXML (or other XML) documents and the proprietary objects of the CFOWeb System. Connect Messaging Server 90 provides centralized (within the CFOWeb 40 System) messaging and conversion functionality, while Connect Processor 20 provides distributed messaging and conversion functionality at Member and Provider client sites. Therefore, in the present embodiment of this invention, descriptions of the messaging and 45 conversion functionality of Connect Processor 20 are also applicable to Connect Messaging 30 Server 90.

5 a. Functional Overview

FIG. 7 illustrates an overview of the Connect Processor and its functionality.

Connect Processor 1010 (including Connect Messaging Server) serves as an intermediary between the CFOWeb System 1000, including its various servers (as shown in FIG. 1), and the systems of Members and Providers. Connect Processor 1010 processes "messages" and "trades." Messages include communications between Members/Providers and the various servers of CFOWeb System 1010 (e.g., chat, e-mail, reports, portfolio management, etc.) that describe actions and events to be performed. Messages include trade information regarding financial transactions between Members and Providers. Note, however, that not all messages include information regarding specific financial transactions.

Members and Providers send requests for price quotes, price quotes, and other messages via an automated message broker 1150, which in turn sends such information through automated connection 1140 to a messaging middleware client application 1130 that is in communication with Connect Processor 1010. Messaging middleware client application 1130 sends the information, in the form of XML streams 1120 to Connect Processor 1010.

Connect Processor 1010 converts the XML information into "Connect" message objects (including trade objects) 1105 (as will be described below). Connect Processor 1010 processes 1070 the message objects 1105 and, if related to trades, sends the message objects 1105 to the CFOWeb System 1000, including the content 1060 provided by the Member or Provider. Alternatively, if the message objects 1105 do not include information regarding specific financial transactions and relate to non-trade functions on CFOWeb System 1000, Connect Processor 1010 will send the message objects 1105 as actions or events to be performed at one of the system servers.

Connect Processor 1010 processes 1070 messages 1050 (which may include trade information) to Members or Providers by converting them into message objects 1075. In addition, Connect Processor 1010 processes actions and events 1030 occurring at any of the system servers by converting them into message objects 1075. Next, Connect Processor 1010 converts 1090 the message objects 1075 into XML documents 1110 (which may be in the form of FinXML documents). Connect Processor 1010 sends the resulting XML documents 1110 (e.g., a price quote or price quote request) to messaging middleware client application 1130. Messaging middleware client application 1130 sends the XML documents 1110 to the automated message broker 1150 of the appropriate Member or Provider through automated

connection 1140, for conversion into objects. Note that in parallel to the processing and conversion of messages and objects from CFOWeb System 1000, Connect Processor 1010 routes the appropriate destination 1020 and addressing information 1080 for the particular Member or Provider that will receive the XML documents 1110. The XML documents (which may be in the form of FinXML documents) will be converted into objects appropriate for processing by the Member or Provider (as described below).

b. Architecture

FIG. 8 shows the architecture of the Connect Processor 3275 in an embodiment of this invention. CFOWeb System 3280 includes Outbound Queue 3200 and Inbound Queue 3205 for the storage of outgoing messages 3210 and incoming messages 3270, respectively. In this embodiment, messages 3210 and 3270 are in "Java Messaging Server" ("JMS") format. Connect Processor 3275 includes Dispatcher module 3215, which extracts the message "payload" 3220 from message 3210 and passes the payload 3220 as a Java object to the appropriate Message Handler 3225. Payload 3220 contains the information represented by the FinXML "Trade" element (described above and in FIG. 3), including information regarding the parties engaged in the transaction and the type of transaction.

Connect Processor 3275 contains one or more Message Handlers 3225; a different Message Handler 3225 can be constructed to handle each type of message to be received by the Member or Provider. Using payload 3220, the appropriate Message Handler 3225 will invoke actions 3230 on the target Member or Provider system 3235, where the action is based on the information contained in payload 3220. The Member/Provider system 3235 communicates with Message Handler 3225 by sending a synchronous response 3240. The Member/Provider system 3235 sends an asynchronous response 3245 to Message Constructor Servlet 3250. Message Constructor Servlet 3250 enables the Member/Provider system 3235 to asynchronously construct messages for the CFOWeb System 3280 by sending parameters via transfer protocol (e.g., HTTP/IP) calls. Message Constructor Servlet 3250 will send the asynchronous message 3255 to Message Sender Service 3265. Message Sender Service 3265 also receives synchronous messages 3260 from Message Handler 3225. Message Sender Service 3265, in turn, forwards the messages 3270 to Inbound Queue 3205 of CFOWeb System 3280.

c. Message Structure

FIG. 9 shows the structure of the messages 1600 that are distributed by the Connect Processor between the CFOWeb System and systems of Members and Providers, in an embodiment of this invention. The system uses the messages to communicate all system events and transactions among system users. There are two categories of messages: “Workflow” messages and “Control” messages. Workflow messages are the main messages that describe the structure and value of transactions, deliver information to and from system servers for portfolio management, trading, and other functions, and deliver information between Members and Providers. Control messages communicate acknowledgement and exception information.

In this embodiment, each message 1600 is expressed in XML in Java Messaging Server” (“JMS”) format. Each message 1600 consists of JMS-based middleware 1610 and document 1620. Middleware 1610, which may be an off-the-shelf product, includes communications protocol (e.g., HTTP/IP, SSL, TCP) and message administration and logging functionality that enable the reliable transmission of XML documents across networks and between the CFOWeb System and the Connect Processor.

Document 1620, which is an XML document, includes header 1630 and message detail 1660. Header 1630, in turn, includes message identification 1640 and routing information 1650. Message identification 1640 includes the message type (e.g., Workflow or Control), a message identifier, and a date/time stamp. Routing information 1650 identifies the message source and destination. Such information is managed by a routing table within the CFOWeb System that maps source and destination identifiers against participating Members and Providers.

Message detail 1660 includes text describing the purpose and detail of the message
25 and may contain the payload 1670 , which includes FinXML Trade information 1680
(represented by the FinXML “Trade” element described above and in FIG. 3) that defines the
transaction.

i. XML Message Structure

FIG. 10 illustrates the structure of a Connect message, as expressed in XML, in the present embodiment of this invention.

5 (a) Message Root Tag

Message root tag 1700 (or "CFOWeb Connect" root tag) identifies the message as a Connect message, and includes the following attributes:

- "System Name": the name of the system that generated the message, e.g., "CFOWeb", "Connect" (for a Member or Provider system), or the name of a third-party system, if applicable.
- "System ID": the identifier of the system that generated the message.
- "Version": the version of the Connect message vocabulary; may differ for different Member/Provider configurations.
- "Test": identifier of messages as "test" ("Y") or "live" ("N"); a test message in a live environment will be communicated but not included and acted on in the business workflow.

In the present embodiment of this invention, the Message root tag 1700 has the following XML definition:

```
<!ELEMENT Message (header, (workflowMsg | controlMsg ) )>
<!ATTLIST Message
    systemName CDATA #REQUIRED
    systemId CDATA #REQUIRED
    version CDATA #FIXED '1.0'
    test (Y | N ) #REQUIRED >
```

25 (b) Header

Header element 1705 describes message identification information, and includes the following attributes:

- "Conversation ID": a system-assigned sequence number that identifies the message as belonging to a particular conversation initiated by one of the communicating parties.

- 5 • “Sequence ID”: a sequence number generated separately by each communicating node that is used as a reference by control messages and to provide chronological ordering of messages.
- 10 5 • “Sent Time”: a system-assigned timestamp which indicates the time that the XML document was formed.

15 In the present embodiment of this invention, the Header element 1705 has the following XML definition:

10 <!ELEMENT header (routing)>
20 <!ATTLIST header conversationId CDATA #REQUIRED
 sequencId CDATA #REQUIRED
 sentTime CDATA #REQUIRED >

15 25 (c) **Routing Information**

20 Routing element 1710 contains reference routing information about the source and destination of the message. This information includes the system-defined identifier of Members and Providers. The routing information is used to derive the middleware-addressing scheme (e.g., point-to-point message queue, topic of a publish/subscribe channel) and to identify the user responsible for the conversation. Routing element 1710 includes the following sub-elements:

- 35 • “Source” 1715: the identifier of the source organization; this is a reference to a Counterparty element; can be anonymous.
- 40 • “Destination” 1720: the identifier of the destination organization; this is a reference to a Counterparty element; can be anonymous.

45 In the present embodiment of this invention, the Routing element 1710 has the following XML definition:

5 5
 <!ELEMENT routing (source , destination)>
 <!ELEMENT source (#PCDATA)>
 <!ELEMENT destination (#PCDATA)>

10

5

(d) Workflow Messages

Workflow Message element 1725 contains descriptions of messages that effect state
transition and actions in the workflow cycle, including financial transactions,
communications between Members and Providers, and interactions with CFOWeb System
servers. Workflow Message element 1725 contains "Note" element 1730, which is used as an
indicator whenever a Member or Provider desires to attach freeform, textual information with
trade information. In addition, each instance of Workflow Message element 1725 contains
one of the following Workflow Message types:

- (1) Quote Request
- (2) Quote Response
- (3) Quote Indicate Interest
- (4) Quote Accept
- (5) Quote Reject
- (6) Withdraw Indication of Interest ("IOI")
- (7) Withdraw Quote Request
- (8) Withdraw Quote
- (9) Withdraw All Quotes
- (10) Disclose
- (11) Price Request
- (12) Price Response
- (13) Quote Request Expiry
- (14) Quote Expiry

Each Workflow Message type element represents a different type of Workflow Message,
which will be described below.

45 30 In the present embodiment of this invention, Workflow Message element 1725 has the
 following XML definition:

50

55

```
<!ELEMENT workflowMsg (note?,>  
(quoteRequest | quoteResponse | quoteIndicateInterest | quoteAccept | quoteReject | withdrawOI |  
withdrawQuoteRequest | withdrawQuote , withdrawAllQuotes | disclose | priceRequest | priceResponse |  
quoteRequestExpiry | quoteExpiry ) >
```

(1) Quote Request Message

Quote Request Message element 1755 describes a message to notify a Provider's system that a Member is requesting a price quote. Quote Request Message element 1755 includes the FinXML trade object as its payload, as well as information regarding the type of quote requested by the Member (e.g., spread). The CFOWeb System may handle an incoming Quote Request Message element 1755 in the following ways: (i) use Provider-configured automated pricing and send a "Quote Response Message" containing a computed price; or (ii) pass the Quote Request information to an internal trading environment to alert a Provider that the quote is available to be filled, in which case the trade details from the payload could be loaded into a back-end spreadsheet or other pricing system to allow a Provider to price the trade manually.

Quote Request Message element 1755 includes the following sub-elements and attributes:

- “Quote Variable” 1760: the variable(s) necessary to express a quote.
 - “Request ID”: identifier of the Quote Request.
 - “Expiry Time”: deadline (in 24-hour format) specified by Member for submission of quotes in response to Quote Request.
 - “Leg Ref”: identifier of particular trade leg for which quote requested, if applicable (e.g., “Leg ID” of particular leg or “None”).
 - “Payload” 1740: information describing a particular financial transaction.
 - “Payload Type”: the category of payload (e.g., FinXML).

5 In the present embodiment of this invention, Quote Request Message element 1755
 has the following XML definition:

10 5 <!ENTITY % payloadDef "payload
 | payloadType">
 <!ELEMENT quoteRequest (quoteVariable+, (%payloadDef;))>
 <!ATTLIST quoteRequest requestId CDATA #REQUIRED
 expiryTime CDATA #REQUIRED >

15 10 The following is an example Quote Request Message element 1755 in the present
 embodiment of this invention:

20 20 <?xml version="1.0"?>
 <!DOCTYPE cfoWebConnect SYSTEM "CFOWEBConnect.dtd">
 <cfoWebConnect systemName="CFOWeb Connect" systemId="cfoweb" version="1.0" test="N">
 15 <header conversationId="000001" sequenceId="000002" sentTime="1999-12-13T19:39:34">
 <routing>
 25 <source>ABC Corp.</source>
 <destination>XYZ</destination>
 </routing>
 20 </header>
 30 <workflowMsg>
 <note>This is a quote request</note>
 <quoteRequest requestId="1234" expiryTime="1999-12-13T19:40:34">
 35 <quoteVariable legRef="none">
 25 <key>fxRate</key>
 </quoteVariable>
 <payloadType="FinXML"/>
 </quoteRequest>
 40 </workflowMsg>
 30 </cfoWebConnect>

45 (2) Quote Response Message

Quote Response Message element 1765 describes a message to notify the CFOWeb

35 System that a Provider has submitted a price quote in response to a Quote Request Message

50

5 from a Member. Quote Response Message element 1765 includes the value of the quoted variables and can optionally include a payload of the complete trade, which is useful where the Provider may have suggested a modified or alternate structure. The CFOWeb System uses the payload information to update the original quote request with a price quote and
10 5 refreshes the requesting Member's web browser to display the offered price quote.

Quote Response Message element 1765 includes the following sub-elements and attributes:

- “Quoted Variable” 1770: the quoted variable(s) used to express a quote.
- 10 • “Key” 1775: name of the quoted variable.
- 20 • “Value” 1780: the value of the price quote.
- 25 15 • “Pricing Detail” 1785: additional information regarding the price quote (e.g., price sensitivity).
- 30 • “Key” 1790: name of the pricing detail.
- 35 20 • “Value” 1795: the value of the pricing detail.
- 40 25 • “Request ID”: identifier of the Quote Request for which Quote Response is submitted.
- 45 30 • “Quote ID”: identifier of the Quote Response.
- 50 25 • “Expiry Time”: deadline (in 24-hour format) specified by Provider for validity of price quote.
- 55 • “Leg Ref”: identifier of particular trade leg for which price quote submitted, if applicable (e.g., “Leg ID” of particular leg or “None”).
- 50 • “Payload” 1740: information describing a particular financial transaction.

5 • “Payload Type”: the category of payload (e.g., FinXML).

In the present embodiment of this invention, Quote Response Message element 1765

has the following XML definition:

```
10      5   <!ELEMENT quotedVariable (%keyValuePair; )>
          <!ATTLIST quotedVariable legRef CDATA #REQUIRED>
          <!ELEMENT pricingDetail (%keyValuePair; )>
          <!ATTLIST pricingDetail legRef CDATA #REQUIRED>
          <!ENTITY % requestQuoteRef " requestId CDATA #REQ
15      10      quotedId CDATA #REQUIRED">
          <!ELEMENT quoteResponse (quotedVariable- , pricingDe
          <!ATTLIST quoteResponse %requestQuoteRef;
              expiryTime CDATA #REQUIRED >
```

15 The following is an example Quote Response Message element 1765 in the present embodiment of this invention:

25

30

35

40

50

```

5           <?xml version="1.0"?>
5           <!DOCTYPE cfoWebConnect SYSTEM "CFOWEBConnect.dtd">
10          <cfoWebConnect systemName="CFOWeb Connect" systemId="connect" version="1.0" test="N">
10              <header conversationId="000001" sequenceId="000005" sentTime="1999-12-13T19:39:52">
5                  <routing>
10                      <source>XYZ</source>
10                      <destination>ABC Corp.</destination>
5                  </routing>
10              </header>
15              <workflowMsg>
10                  <note>This is a quoteResponse</note>
20                  <quoteResponse requestId="1234" quoteId="1" expiryTime="1999-12-13T19:40:22">
15                      <quotedVariable legRef="none">
20                          <key>fxRate</key>
15                          <value>102</value>
25                      </quotedVariable>
20                      <pricingDetail legRef="none">
15                          <key>market data</key>
25                          <value>Reuters at 1999-12-13T19:41:09</value>
20                  </pricingDetail>
30                  </quoteResponse>
30              </workflowMsg>
</cfoWebConnect>

```

25

35 (3) Other Workflow Messages

In the present and other embodiments of this invention, Workflow Message element 1725 can include other message types to enable communications related to financial transactions.

40

30 (i) Quote Indicate Interest Message

Quote Indicate Interest Message element 1800 describes a message used by the CFOWeb System 3280 (in FIG. 8) to notify the Connect Processor 3275 that a Member has indicated interest in a price quote submitted by a Provider in response to the Member's earlier quote request. The Connect Processor 3275 can be configured with a Message Handler 3225

50

5 that will route Quote Indicate Interest Message element 1800 to the Provider's internal system 3235 as a screen pop-up or alert.

(ii) **Quote Accept Message**

10 5 Quote Accept Message element 1805 describes a message used by the CFOWeb System to notify the Connect Processor that a Member wishes to accept the price quote submitted by a Provider. Quote Accept Message element 1805 includes a reference to the quote request and the price accepted by the Member. The system will send the Quote Accept
15 Message only to the Provider whose price was accepted; all other Providers who submitted
10 price quotes in response to the quote request will receive a "Quote Reject Message"
(described below). The Connect Processor 3275 (in FIG. 8) can be configured with a
20 Message Handler 3225 that will route Quote Accept Message element 1805 to the Provider's
internal system 3235 as a screen pop-up or alert.

25 15 (iii) **Quote Reject Message**

25 15 Quote Reject Message element 1810 describes a message used by the CFOWeb System to notify a Provider that a Member has rejected the price quote submitted by the Provider. This will occur when a Member expressly rejects a Provider's price quote, or
30 accepts another Provider's quote in response to the same quote request, thus implicitly
20 rejecting all other price quotes. Quote Reject Message element 1810 includes a reference to the quote request. The Connect Processor 3275 (in FIG. 8) can be configured with a
35 Message Handler 3225 that will route Quote Reject Message element 1810 to the Provider's
internal system 3235 as a screen pop-up or alert.

40 25 (iv) **Withdraw Indication of Interest
Message**

45 25 Withdraw Indication of Interest ("IOI") Message element 1815 describes a message used by the CFOWeb System 3280 (in FIG. 8) to notify the Connect Processor 3275 that a
30 Member has withdrawn its indication of interest in a price quote submitted by a Provider in response to the Member's earlier quote request. The Connect Processor 3275 can be
50

5 configured with a Message Handler 3225 that will route WithdrawOI Message element 1815
to the Provider's internal system 3235 as a screen pop-up or alert.

(v) Withdraw Quote Request Message

10 5 Withdraw Quote Request Message element 1820 describes a message used by the
CFOWeb System to notify the Connect Processor that a Member wishes to withdraw a quote
request that was sent previously. All Providers that were sent the original Quote Request
15 Message will receive the Withdraw Quote Request Message as they no longer need to track
activity on their price quotes regarding the particular quote request. its indication of interest
in a price quote submitted by a Provider in response to the Member's earlier quote request.
20 10 The Connect Processor 3275 (in FIG. 8) can be configured with a Message Handler 3225 that
will route Withdraw Quote Request Message element 1820 to the Provider's internal system
3235 as a screen pop-up or alert.

25 15 (vi) Withdraw Quote Message

Withdraw Quote Message element 1825 describes a message used by the CFOWeb
System to indicate that a Provider wishes to withdraw a price quote that was sent previously.
30 The Withdraw Quote Message can be sent from either the CFOWeb System if a Provider
withdraws the price quote manually or through the Connect Processor if the withdrawal
20 action is generated by means of a Provider's internal system (either manually or
automatically). If the Withdraw Quote Message is generated through the Connect Processor,
35 a synchronized clock timestamp will be set on the message indicating the expiration time of
the price quote.

40 25 (vii) Disclose Message

Disclose Message element 1830 describes a message used by the CFOWeb System to
disclose to a party the identity of a previously undisclosed Counterparty. Such disclosure will
only occur upon notification of the system by the Counterparty to disclose its identity.

45 30 (viii) Price Request Message

Price Request Message element 1835 describes a message used by the CFOWeb
System for semi-automated pricing to notify the Connect Processor that a Member is

5 requesting a price quote for a request from the Member's internal system. Price Request
Message element 1835 includes the FinXML trade object as its payload, as well as
information regarding the type of quote requested by the Member (e.g., spread). The Connect
Processor handles the message with one or more Providers and sends the CFOWeb System a
10 5 "Price Response Message" (described below) containing a price quote.

(ix) Price Response Message

15 Price Response Message element 1840 describes a message used by the Connect
Processor for semi-automated pricing to notify the CFOWeb System that a Provider's internal
10 system has calculated a price quote for a quote request and to submitted the price quote to the
CFOWeb System. The CFOWeb System uses the information to refresh the requesting
20 Member's web browser to display the offered price quote. The Provider may submit the
quote with this pricing information or with information entered manually. In either case, the
Provider submits the price quote to the Member manually (e.g., by clicking a button).

25 15

(x) Quote Request Expiry Message

30 Quote Request Expiry Message element 1845 describes a message used by the
CFOWeb System to notify the Connect Processor that a Member's quote request has expired.
The CFOWeb System generates the Quote Request Expiry Message automatically upon the
20 occurrence of the expiry time for the quote request. All Providers that were sent the original
Quote Request Message will receive the Quote Request Expiry Message as they no longer
35 need to track activity on their price quotes regarding the particular quote request. The
Connect Processor 3275 (in FIG. 8) can be configured with a Message Handler 3225 that will
route Quote Request Expiry Message element 1845 to the Provider's internal system 3235 as
25 a screen pop-up or alert.

40

(xi) Quote Expiry Message

45 Quote Expiry Message element 1850 describes a message used by the CFOWeb
System to notify the Connect Processor that a Provider's price quote has expired. The
30 CFOWeb System generates the Quote Expiry Message automatically upon the occurrence of
the expiry time for the price quote.

50

5 (xii) Withdraw All Quotes Message

Withdraw All Quotes Message element 1855 describes a message used by the
CFOWeb System to notify the Connect Processor that a Provider wishes to withdraw all price
quotes. The message can specify criteria for the quotes to be withdrawn.

10 15 (e) Control Messages

Control Message element 1860 contains descriptions of messages that are sent in
response to Workflow Messages to indicate the success or failure of message receipt and
processing. While the middleware serves to transmit messages between the CFOWeb System
and the Connect Processor, the middleware does not guarantee certain system performance
parameters, including particular delivery time, successful translation and processing of the
XML content, or the successful provision of a price quote. Thus, Control Message element
1860 provides acknowledgement of message delivery and reports error conditions to the
sender of a message.

20 25 30 Control Message element 1860 includes a "Sequence ID" element, which is a system-
assigned sequence number for the particular Workflow Message to which Control Message
element 1860 applies. In addition, each instance of Control Message element 1860 contains
one of the following Control Message types:

(1) Ack

20 (2) Error

35 Each Control Message type element represents a different type of Control Message, which
will be described below.

In the present embodiment of this invention, Control Message element 1860 has the
following XML definition:

40 45 <!ELEMENT controlMessage ack | error>
<!ATTLIST controlMessage sequenceld CDATA #REQUIRED >

(I) Acknowledge Message

30 35 40 Acknowledge ("Ack") Message element 1865 is used to acknowledge the successful
receipt, translation, and processing of a Connect message and transaction payload. Ack
Message element 1865 includes "Our Payload Ref" element 1870, which contains a reference

5 to a Payload element 1740 carried by the acknowledged message. Our Payload Ref element
1870 includes the following sub-elements:

- “Payload Type”: the category of payload (e.g., FinXML).
- “Payload ID”: the identifier of a previously communicated payload.

10 In the present embodiment of this invention, Ack Message element 1865, including
Our Payload Ref element 1870, has the following XML definition:

```
<!ENTITY % payloadRef " payloadType CDATA #REQUIRED
    payloadId CDATA #REQUIRED">
<!ELEMENT ourPayloadRef EMPTY>
<!ATTLIST ourPayloadRef %payloadRef; >
<!ELEMENT ack (ourPayloadRef? )>
```

20 The following is an example Ack Message element 1865 in the present embodiment
of this invention:

```
<?xml version="1.0"?>
<!DOCTYPE cfoWebConnect SYSTEM "CFOWEBConnect.dtd">
<cfoWebConnect systemName="CFOWeb Connect" systemId="connect" version="1.0" test="N">
    <header conversationId="000001" sequenceId="000003" sentTime="1999-12-13T19:39:52">
        <routing>
            <source>ABC Corp.</source>
            <destination>XYZ</destination>
        </routing>
    </header>
    <controlMsg sequenceId="000001">
        <ack/>
    </controlMsg>
</cfoWebConnect>
```

40 In the present and other embodiments of this invention, Ack Message element 1865
30 may include specific acknowledgement messages for verification and completion of a
transaction, as described below.

45

50

55

5

(i) Trade Download Response Message

10

Trade Download Response Message element describes a message used by the CFOWeb System to notify a Provider's internal system that both the Provider and a Member have agreed to the terms of a particular price quote and that the specified trade should now be processed. The Connect Processor uses the Trade Download Response Message element to send all relevant trade information to the Provider's internal system for processing. The Trade Download Response Message element includes the trade payload.

15

10

(ii) Trade Download Acknowledge Message

20

Trade Download Acknowledge Message element describes a message used by the CFOWeb System to notify the Connect Processor that all necessary internal systems of the Provider have completed initial processing for a particular trade.

25

15

(iii) Trade Download Request Message

30

Trade Download Request Message element describes a message used by the Connect Processor when it needs to download executed trades from the CFOWeb System. Typically, this occurs when trades did not load properly. The CFOWeb System uses the Trade Download Request Message to send all trades to the Connect Processor so that it may process and feed the trade information to Providers' internal systems.

40

25

(iv) Deal Verify Request Message

45

Deal Verify Request Message element describes a message used by the Connect Processor to notify the CFOWeb System that a completed transaction has been verified at the Provider internal system and to request that the CFOWeb System also verify the completed transaction.

30

50

55

5 (v) Deal Verify Acknowledge Message

Deal Verify Acknowledge Message element describes a message used by the Connect Processor to communicate confirmation to the CFOWeb System that a Deal Verify Request Message has been received.

10

5

(vi) Deal Verify Confirm Message

Deal Verify Confirm Message element describes a message used by the CFOWeb System to communicate confirmation to the Connect Processor that a verification request has been carried out successfully.

15

10

(2) Error Message

Error Message element 1875 is used to provide notification to the sender of a message any time application-level processing of the XML message content fails, including the unsuccessful translation of XML into objects or execution of a pricing algorithm. Error Message element 1875 includes the following sub-elements:

- “Error Code” 1880: the identifier of the particular type of error.
- “Error Text” 1885: the text description of the particular type of error.

In the present embodiment of this invention, Error Message element 1875, has the following XML definition:

20 <!ELEMENT error (errorText?, errorCode)>
<!ELEMENT errorText (#PCDATA)>
35 <!ELEMENT errorCode (#PCDATA)>

The following is an example Error Message element 1875 in the present embodiment

25 of this invention:

40

45

50

```

5      <?xml version="1.0"?>
<!DOCTYPE cfoWebConnect SYSTEM "CFOWEBConnect.dtd">
10     <cfoWebConnect systemName="CFOWeb Connect" systemId="connect" version="1.0" test="N">
          <header>
15          <routing>
              <source>ABC Corp.</source>
              <destination>XYZ</destination>
10          </routing>
          <message payloadType="FinXML" payloadId="123456" sequenceId="000005"
sentTime="1999-12-13T19:39:22">
              <error sequenceId="000001">
                  <errorText>Failed to instantiate trade in Connect Cache</errorText>
                  <errorCode>001</errorCode>
              </error>
15          </message>
          </header>
          <body>
20             <note>This is an error control message</note>
          </body>
25         </cfoWebConnect>
20

```

30

d. Message Flow

35 25 The flow of Workflow Messages back and forth from the CFOWeb System through
 the Connect Processor to Member and Provider internal systems differs depending on the
 type of Workflow Message (e.g., quote request, price quote) and the type of processing (e.g.,
 automated, manual, synchronous, asynchronous).

40

i. Automated Pricing – Synchronous

45 FIG. 11 illustrates the flow of Workflow Messages when synchronous automated
 pricing occurs. CFOWeb System 3280 sends Quote Request Message 3310 from Outbound
 Queue 3200 to Dispatcher module 3215 in Connect Processor 3275. Dispatcher 3215 extracts
 the payload from Quote Request Message 3310 and passes the payload as a Trade Object
50 35 (Java object) 3315 to the Quote Request Message Handler 3305. Using the payload in Trade

5 Object 3315, Quote Request Message Handler 3305 executes a “Call Price Function” 3320 on
the target Provider pricing engine 3300 in the Provider’s internal system. Call Price Function
3320 notifies the Provider’s pricing engine 3300 to calculate and send a price quote, based on
the information contained in Trade Object 3315. The Provider’s pricing engine 3300 sends a
10 5 synchronous response back to Quote Request Message Handler 3305 in the form of a “Return
Price” Message 3325. Quote Request Message Handler 3305 generates a Quote Response
Message 3330 using the price quote and sends it to Message Sender Service 3265. Message
Sender Service 3265, in turn, forwards the Quote Response Message 3335 to Inbound Queue
15 3205 of CFOWeb System 3280 for processing.

10

ii. Automated Pricing – Asynchronous

20 FIG. 12 illustrates the flow of Workflow Messages when asynchronous automated
pricing occurs. CFOWeb System 3280 sends Quote Request Message 3310 from Outbound
Queue 3200 to Dispatcher module 3215 in Connect Processor 3275. Dispatcher 3215 extracts
25 15 the payload from Quote Request Message 3310 and passes the payload as a Trade Object
(Java object) 3315 to the Quote Request Message Handler 3305. Using the payload in Trade
Object 3315, Quote Request Message Handler 3305 executes a “Call Price Function” 3320 on
the target Provider pricing engine 3300 in the Provider’s internal system. Call Price Function
3320 notifies the Provider’s pricing engine 3300 to calculate and send a price quote, based on
30 20 the information contained in Trade Object 3315. The Provider’s pricing engine 3300 sends
an asynchronous response that contains message details 3328 (*i.e.*, price quote) to Message
Constructor Servlet 3250. Message Constructor Servlet 3250 constructs a Quote Response
35 30 Message 3330 using the price quote and sends it to Message Sender Service 3265. Message
Sender Service 3265, in turn, forwards the Quote Response Message 3335 to Inbound Queue
40 25 3205 of CFOWeb System 3280 for processing.

iii. Semi-Automated Pricing – Synchronous

45 FIG. 13 illustrates the flow of Workflow Messages when synchronous semi-
automated pricing occurs. CFOWeb System 3280 sends Quote Request Message 3310 and
50 30 Price Request Message 3340 from Outbound Queue 3200 to Dispatcher module 3215 in
Connect Processor 3275. Price Request Message 3340 is a message used by the CFOWeb
System 3280 for semi-automated pricing to notify the Connect Processor 3275 that a Member

5 is requesting a price quote for a request from the Member's internal system. Dispatcher 3215
extracts the payload from Quote Request Message 3310 and passes the payload as a Trade
Object (Java object) 3315 to the Price Request Message Handler 3400. Using the payload in
Trade Object 3315, Price Request Message Handler 3400 executes a "Call Price Function"
10 5 3320 on the target Provider pricing engine 3300 in the Provider's internal system. Call Price
Function 3320 notifies the Provider's pricing engine 3300 to calculate and send a price quote,
based on the information contained in Trade Object 3315.

15 The Provider's pricing engine 3300 sends a synchronous response back to Price
Request Message Handler 3400 in the form of a "Return Price" Message 3325. Price Request
10 10 Message Handler 3400 generates a Price Response Message 3345 using the price quote and
sends it to Message Sender Service 3265. Price Response Message 3345 is a message used
20 20 by the Connect Processor 3275 for semi-automated pricing to notify the CFOWeb System
3280 that a Provider's internal system has calculated a price quote for a quote request and to
submitted the price quote to the CFOWeb System 3280; the CFOWeb System 3280 uses the
25 15 information to refresh the requesting Member's web browser to display the offered price
quote. Message Sender Service 3265, in turn, forwards the Price Response Message 3350 to
Inbound Queue 3205 of CFOWeb System 3280 for processing.

30 iv. **Deal Transmission – Asynchronous**

20 FIG. 14 illustrates the flow of Workflow Messages when asynchronous transmission
of a completed transaction occurs. CFOWeb System 3280 sends Trade Download Response
35 35 Message 3510 from Outbound Queue 3200 to Dispatcher module 3215 in Connect Processor
3275. Trade Download Response Message is a message used by the CFOWeb System 3280
to notify a Provider's internal system that both the Provider and a Member have agreed to the
40 25 terms of a particular price quote and that the specified trade should now be processed. The
Connect Processor uses the Trade Download Response Message to send all relevant trade
information to the Provider's internal system (i.e., deal capture system 3505) for processing.

45 Dispatcher 3215 extracts the payload from Trade Download Response Message 3510
and passes the payload as a Trade Object (Java object) 3315 to the Trade Download Response
30 30 Message Handler 3500. Using the payload in Trade Object 3315, Trade Download Response
Message Handler 3500 executes a "Call Deal Capture Function" 3515 on the target Provider
50 35 deal capture system 3505 in the Provider's internal system. Call Deal Capture Function 3515

5 notifies the Provider's deal capture system 3505 to process the completed transaction, based
on the information contained in Trade Object 3315. The Provider's deal capture system 3505
sends an asynchronous response containing message details 3520 to Message Constructor
Servlet 3250. Message Constructor Servlet 3250 constructs a Trade Download Acknowledge
10 5 ("Ack") Message 3525 using message details 3520 and sends it to Message Sender Service
3265. Trade Download Ack Message is a message used by the Connect Processor 3275 to
notify the CFOWeb System 3280 that all necessary internal systems of the Provider have
15 completed initial processing for a particular trade. Message Sender Service 3265, in turn,
forwards the Trade Download Ack Message 3530 to Inbound Queue 3205 of CFOWeb
20 10 System 3280 for processing.

20 3. "FinScript"

The present invention enables users (Members and Providers) to conduct financial
transactions using the CFOWeb System and Connect Processor via connections to the users'
25 15 internal, back-end systems. In the present embodiment of this invention, the Connect
Processor enables the communication of information related to financial transactions between
users (*i.e.*, Members and Providers) and the CFOWeb System by converting FinXML (or
30 20 other XML) documents to/from proprietary financial (Java) objects – as used on the users'
internal systems – using proprietary stylesheets created in XSL, known as "FinScript". The
25 30 Connect Processor 20 (as shown in FIG. 1) creates a FinXML document that can be sent
using a transfer protocol (*e.g.*, HTTP/IP) to the Connect Messaging Server 90 for conversion
35 40 to objects that can be processed on the server side. Following processing, the Connect
Messaging Server 90 converts the objects to a FinXML (or other XML) document, using
XSL stylesheets, and sends the FinXML (or other XML) document to the Connect Processor
25 40 45, which uses FinScript to create a JavaScript program from the FinXML (or other XML)
document. In turn, Java objects are created from the JavaScript program and sent to the other
organization (*e.g.*, a Provider).

45 a. Conversion (Encoding) of Financial Objects to FinXML
30 Documents

5 When a user (Member or Provider) wishes to send information (e.g., a quote request
or a price quote) to the CFOWeb System, the Connect Processor must convert the proprietary
financial objects used by the user's internal system into FinXML (or other XML) documents
that can be used by the CFOWeb System. FIG. 15 illustrates the components of the
10 5 conversion (or encoding) process and FIG. 16 shows the steps to be executed by the system
to perform such conversion, in an embodiment of the present invention. Note that these steps
could be combined, certain steps could be removed and others deleted, and/or the order of the
15 steps could be modified, in various other embodiments of this invention.

When the user wishes to submit information regarding a transaction (e.g., a quote
10 request from a Member, a price quote from a Provider), the user's messaging client sends the
financial objects 1400 (as shown in FIG. 15) as represented on the user's internal system to
the Connect Processor via an application programming interface ("API") (step 1470 of FIG.
20 16). Typically, financial objects 1400 will be stored on the user's internal system as Java
objects, which are in the form of "object graphs." Such object graphs consist of inter-linked
25 15 nodes representing the elements and attributes of the financial object.

Upon receiving financial objects 1400, the Connect Processor will identify the
30 applicable XML object mapping 1410 to apply to financial objects 1400 (step 1480). In some
embodiments of this invention, XML object mappings 1410 may be customized by the user,
in order to correspond to the form and structure of the user's proprietary financial objects.

35 20 The following is an example XML object mapping 1410 used in the present
embodiment of this invention:

40

45

50

```
5 <object class='com.integral.finance.fx.FXRateC' tag='fxRate'>
 6   <objectProperty tag='baseQuoteCcy' accessor='getBaseQuoteCcy' />
 7   <doubleProperty tag='rate' accessor='getRate' />
 8   <objectProperty tag='variableQuoteCcy' accessor='getVariableQuoteCcy' />
 9
10 5 </object>
11
12 <object class='com.integral.finance.currency.CurrencyC' tag='currency'>
13   <stringProperty tag='isoCode' accessor='getISOName' />
14 </object>
15
16 10 <object class='com.integral.finance.fx.FXTradeC' tag='fxTrade'>
17   <objectProperty tag='dealtCcy' accessor='getDealtCcy' />
18   <doubleProperty tag='dealtPrincipal' accessor='getDealtPrincipal' />
19   <objectProperty tag='fxRate' accessor='getFXRate' />
20   <objectProperty tag='settledCcy' accessor='getSettledCcy' />
21   <doubleProperty tag='settledPrincipal' accessor='getSettledPrincipal' />
22   <dateProperty tag='valueDate' accessor='getValueDate' />
23   <booleanProperty tag='isBuy' accessor='isBuy' />
24
25 </object>
```

Next, the Connect Processor invokes a dynamic Document Object Model (“DOM”) parser module 1420 to parse financial objects 1400 and apply XML object mapping 1410 to the elements and attributes of financial objects 1400 (step 1490). DOM is a platform- and language neutral interface that will allow programs and scripts to dynamically access and update the content, structure and style of documents. DOM provides a standard set of objects for representing HTML and XML documents, a standard for how these objects can be combined, and a standard interface for accessing and manipulating them. DOM is described in the [Document Object Model \(DOM\) Level 1 Specification Version 1.0](#) (Oct. 1, 1998), World Wide Web Consortium (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University)
<<http://www.w3.org/TR/REC-DOM-Level-1>>.

The dynamic DOM parser generates a DOM “tree” (1430), which is a 1:1 mapping to the object graph of financial objects 1400 (step 1500). Generation of the DOM tree is dynamic and occurs on an as-needed basis as finite boundaries (transitive closure) of the object graph are determined. Thus, steps 1490 and 1500 may be repeated as necessary. Next

5 the Connect Processor obtains the XSL stylesheet 1440 to apply to DOM tree 1430 (step
1510), based on the object values contained in DOM tree 1430. The proprietary XSL
stylesheet 1440 – known as “FinScript” – contains rules for navigating (i.e., determining
boundaries of) and converting DOM tree 1430 into a FinXML document. In the present
10 embodiment of this invention, XSL stylesheets 1440 are linked to a single root. In some
embodiments of this invention, XSL stylesheets 1440 may be customized by the user, in
order to correspond to the form and structure of the user’s proprietary financial objects.

15 The following is an example XSL stylesheet 1440 used in the present embodiment of
this invention:

```
20 10 <xsl:stylesheet xmlns:xsl="http://www.w3.org/XSL/Transform/1.0">
    <xsl:import href="counterparties2XML.xsl" />
    <xsl:import href="fxUtil2XML.xsl" />
    <xsl:import href="events2xml.xsl" />
    <xsl:output method="xml" indent="yes" />
    <!-- replace the built-in rules for text and attributes -->
    <xsl:template match="text()|@*" />
    <xsl:template name="fxSpot2XML">
        <fxSpot>
            <entryDate>
                <xsl:value-of select="getTradeDate" />
            </entryDate>
            <xsl:apply-templates select="getTradeDate" mode="fxSpot2XML" />
            <xsl:apply-templates select="getSettlementDate" mode="fxSpot2XML" />
            <xsl:apply-templates select="getValueDate" mode="fxSpot2XML" />
            <xsl:apply-templates select="getDealCurrency" mode="fxSpot2XML" />
            <xsl:apply-templates select="getSettledCurrency" mode="fxSpot2XML" />
            <events>
                <xsl:apply-templates select="getFinancialEvents" mode="events2xml" />
            </events>
        </fxSpot>
    </xsl:template>
    <!-- fxSpot2XML -->
</xsl:stylesheet>
```

30 35 Next, the Connect Processor invokes a XSLT processor 1450 – an off-the-shelf
component (e.g., International Business Machines Corp.’s “Alphaworks”) – to apply the rules

5 of the XSL stylesheet 1440 to DOM tree 1430 (step 1520). This process results in the
 generation of a FinXML document 1460 (step 1530) that can be used by the CFOWeb
 System. The following is an example FinXML document 1460 generated by the XSLT
 processor 1450 in the present embodiment of this invention:

```

10 <fxSpot>
  <tradeDate>1999-12-24</tradeDate>
  <valueDate>1999-11-04</valueDate>
  <dealtAmount payer="ABC" receiver="XYZ">
    <currency>JPY</currency>
    <amount>100000000</amount>
    </dealtAmount>
    <settledAmount payer="XYZ" receiver="ABC">      <currency>USD</currency>
      <fxRate>
        <baseCurrency> USD </baseCurrency>
        <baseUnits> 1 </baseUnits>
        <quoteCurrency> JPY </quoteCurrency>
        <quoteUnits> 1 </quoteUnits>
        <rate> 102.5 </rate>
      </fxRate>
    </settledAmount>
  </fxSpot>

```

Note that the same process described above will be used by the Connect Messaging
 Server to convert the proprietary financial objects used by the various servers of the CFOWeb
 System into FinXML (or other XML) documents that can be sent to the Connect Processor.

35 b. **Conversion (Decoding) of FinXML Documents to Financial
 40 Objects**

45 30 When the CFOWeb System is ready to send information regarding a transaction to a
 user (Member or Provider) Connect Processor must convert the FinXML (or other XML)
 documents into proprietary financial objects that can be used by the user's internal system.

FIG. 17 illustrates the components of the conversion (or decoding) process and **FIG. 18**
 50 shows the steps to be executed by the system to perform such conversion, in an embodiment

5 of the present invention. Note that these steps could be combined, certain steps could be removed and others deleted, and/or the order of the steps could be modified, in various other embodiments of this invention.

10 When the CFOWeb System wishes to send information regarding a transaction (e.g., a quote request from a Member, a price quote from a Provider), the Connect Messaging Server sends the previously-created FinXML (or other XML) document 1200 (as shown in FIG. 17) to the Connect Processor (step 1300 of FIG. 18). The following is an example FinXML 15 document 1200 created in the present embodiment of this invention:

```
<fxSpot>
 10   <tradeDate>1999-12-24</tradeDate>
 15   <valueDate>1999-11-04</valueDate>
 20   <dealtAmount payer="ABC" receiver="XYZ">
 25     <currency>JPY</currency>
 30     <amount>100000000</amount>
 35   </dealtAmount>
 40   <settledAmount payer="XYZ" receiver="ABC">    <currency>USD</currency>
 45     <fxRate>
      <baseCurrency> USD </baseCurrency>
      <baseUnits> 1 </baseUnits>
      <quoteCurrency> JPY </quoteCurrency>
      <quoteUnits> 1 </quoteUnits>
      <rate> 102.5 </rate>
    </fxRate>
    </settledAmount>
 50 </fxSpot>
```

Upon receiving FinXML (or other XML) document 1200, the Connect Processor will 40 obtain the XSL stylesheet 1440 to apply to FinXML document 1200 (step 1310), based on the transaction type identified in FinXML document 1200. There is a different XSL stylesheet 45 for each type of transaction and all options supported by the CFOWeb System. The proprietary XSL stylesheet 1210 – known as “FinScript” – contains rules for converting FinXML document 1200 into a JavaScript program, including reusable fragments of JavaScript programming code. In the present embodiment of this invention, XSL stylesheets 50 1210 are linked to a single root. In some embodiments of this invention, XSL stylesheets

5 **1210** may be customized by the user, in order to correspond to the form and structure of the
user's proprietary financial objects.

The following is an example XSL stylesheet **1210** used in the present embodiment of
this invention:

```

10   <xsl:stylesheet xmlns:xsl="http://www.w3.org/XSL/Transform/1.0">
    xmlns="http://www.finxml.org/finxml/1.0"
    <xsl:output method="text" />
    <xsl:output indent="yes" />
15   <xsl:template match="text()|@*" mode="fxSpot" />
10   <xsl:template match="fxSpot">
    <xsl:text>someProperties=newPackages.java.util.HashMap();
    someProperties.put
20     (Packages.com.integral.finance.trade.TradeCreationKeys.TRADE_DATE, "</xsl:text>
    <xsl:value-of select="tradeDate" />
15     <xsl:text>")trade =
    Packages.com.integral.apps.ui.fxtrade.FXTradeFactory.newFXSpotTrade
25     (applicationEnvironment, uow, null, null, someProperties);
    trade.setFrontOfficeID(tradeID);
    </xsl:text>
20   <xsl:apply-templates select="externalId" mode="fxSpot" />
    <xsl:apply-templates select="valueDate" mode="fxSpot" />
    <xsl:apply-templates select="settlementDate" mode="fxSpot" />
    <xsl:apply-templates select="dealtAmount" mode="fxSpot" />
    <xsl:apply-templates select="settledAmount" mode="fxSpot" />
30     events = trade.getFinancialEvents();
    <xsl:apply-templates select="events" mode="events">
        <xsl:with-param name="object" select=""events"" />
        </xsl:apply-templates>
    </xsl:template>
40   <!-- fxSpot -->
30   ...
</xsl:stylesheet>
```

45 Next, the Connect Processor invokes a XSLT processor **1220** – an off-the-shelf
35 component (e.g., International Business Machines Corp.'s "Alphaworks") – to apply the rules
of the XSL stylesheet **1210** to FinXML (or other XML) document **1200** (step **1320**). This

50

5 process results in the generation of a JavaScript program 1230 (step 1330) that can be
executed to generate Java objects. The following is an example JavaScript program 1230
generated by the XSLT processor 1220 in the present embodiment of this invention:

```
10    counterpartyA = Packages.com.integral.finance.counterparty.CounterpartyFactory.newLegalEntity();  
15    ...  
10    someProperties = newPackages.java.util.HashMap();  
15    someProperties.put(Packages.com.integral.finance.trade.TradeCreationKeys.TRADE_DATE, "2000-06-12")  
20    trade = packages.com.integral.apps.ui.fxtrade.FXTradeFactory.newFXSpotTrade(applicationEnvironment,  
25    uow, null, null, someProperties);  
10    valueDate = Packages.com.integral.finance.dateTime.DateTimeFactory.newDate("2000-06-14");  
15    trade.setValuedate (valueDate);  
20    ...  
25    trade.setCounterpartyA (counterpartyA);  
30    trade.setCounterpartyB (counterpartyB);
```

15

25 Next, the Connect Processor invokes a JavaScript interpreter 1240 – an off-the-shelf
component (e.g., Mozilla.org’s “Rhino” JavaScript interpreter) – to execute the JavaScript
program 1230 (step 1340). This process results in the generation of financial objects 1250 –
Java objects – (step 1350) that can be used by the user’s internal systems. The Connect
30 Processor sends the financial objects 1250 to the messaging client application of the user’s
system via an API (step 1360).

35 Note that the same process described above will be used by the Connect Messaging
Server to convert the FinXML (or other XML) documents created and sent by the Connect
Processor into proprietary financial objects to be used by the various servers of the CFOWeb
40 System.

45 The foregoing description of a preferred embodiment of the invention has been
presented for purposes of illustration and description. It is not intended to be exhaustive or to
limit the invention to the precise forms disclosed. Obviously, many modifications and
variations will be apparent to practitioners skilled in this art. One skilled in the art will
readily appreciate that other applications may be substituted for those set forth herein without
departing from the spirit and scope of the present invention. Accordingly, the invention
should only be limited by the Claims included below.

50

Claims

~5

10

15

20

25

30

35

40

45

50

55

CLAIMS

- 5 1. A system for facilitating transactions involving the exchange of financial instruments,
2 the system comprising:
3 (a) a transaction data parser that can extract transaction data describing a
4 transaction conducted using the system, wherein the transaction data can
5 include:
6 (i) a type of the transaction;
7 (ii) a plurality of parties to the transaction;
8 (iii) a payment schedule pursuant to which a first party to the transaction
9 will make one or more payments to a second party to the transaction;
10 and
11 (iv) rate information which can include information relating to an interest
12 rate and/or an exchange rate that can affect the calculation of the
13 amount of one or more of the payments made pursuant to the payment
14 schedule;
15 (b) a reference data parser that can extract profile information about the parties to
16 the transaction; and
17 (c) a market data parser that can extract market data provided to the system, the
18 market data including market interest rates and/or currency exchange rates.
- 20 1. 2. The system of claim 1 wherein the type of the transaction is a foreign exchange spot.
- 25 1. 3. The system of claim 1 wherein the type of the transaction is a foreign exchange
30 forward.
- 35 1. 4. The system of claim 1 wherein the type of the transaction is an interest rate fixed-float
40 swap.
- 45 1. 5. The system of claim 1 wherein the type of the transaction is an interest rate float-float
50 swap.
1. 6. The system of claim 1 wherein the type of the transaction is an interest rate cap.
1. 7. The system of claim 1 in which the type of the transaction is an interest rate floor.

- 5 1 8. The system of claim 1 wherin the type of the transaction is a fixed deposit.
- 10 1 9. The system of claim 1 wherein the type of the transaction is a fixed loan.
- 15 1 10. The system of claim 1 wherein the type of the transaction is a float deposit.
- 20 1 11. The system of claim 1 wherein the type of the transaction is a float loan.
- 25 1 12. The system of claim 1 wherein the type of the transaction is a foreign exchange
 2 option.
- 30 1 13. The system of claim 1 wherein the type of the transaction is a foreign exchange swap.
- 35 1 14. The system of claim 1 wherein the type of the transaction is a cross currency fixed-
 2 fixed swap.
- 40 1 15. The system of claim 1 wherein the type of the transaction is a cross currency fixed-
 2 float swap.
- 45 1 16. The system of claim 1 wherein the type of the transaction is a cross currency float-
 2 float swap.
- 50 1 17. The system of claim 1 wherein the type of the transaction is a forward rate agreement.
- 55 1 18. The system of claim 1 wherein the type of the transaction is a customized trade.
- 60 1 19. A system for facilitating transactions involving the exchange of financial instruments,
 2 the system comprising:
- 65 3 (a) a member client node that can be used by a member user of the system to
 4 conduct a transaction;
- 70 5 (b) a provider client node that can be used by a provider user of the system to
 6 conduct the transaction with the member user; and
- 75 7 (c) a server node that facilitates the exchange of messages among the member
 8 client and provider client nodes, including a transaction message that includes
 9 information describing the transactions between the member user and the
 10 provider user, the server node including:

1 20. The system of claim 19 wherein the messages whose exchange is facilitated by the
15 server node can include:

1 21. The system of claim 19 wherein the negotiated terms include pricing of the
2 transaction.

1 22. The system of claim 19 wherein the negotiated terms include acceptance of the pricing
2 of the transaction.

1 23. The system of claim 19 wherein the negotiated terms include settlement of the
2 transaction.

1 24. The system of claim 19 wherein both the member client node and the provider client
2 node are physically remote from the server node.

- 5 1 25. An automated financial object conversion and messaging system for exchanging
2 messages describing transactions involving the exchange of financial instruments between a
3 provider user of the system and one or more other member users of the system, the system
4 comprising:
10 5 (a) a server node that can convert a first set of internal objects, representing the
6 terms of a transaction between the provider user and a member user of the
7 system, into a transaction message employing a standard format for
8 representing the transaction;
15 9 (b) a provider node that can convert the transaction message into a second set of
10 internal objects; and
20 11 (c) an automated message exchange mechanism that automatically can exchange
12 messages employing the standard format, including the transaction message,
13 between the server node and the provider node.
- 25 1 26. The system of claim 25 wherein the standard format is an XML format.
- 30 1 27. The system of claim 26 wherein the XML format can represent:
2 (a) transaction data describing the transaction, including:
3 (i) a type of the transaction;
4 (ii) a plurality of parties to the transaction, including the provider user and
5 the member user of the system;
6 (iii) a payment schedule pursuant to which the member user will make one
7 or more payments to the provider user; and
8 (iv) rate information which can include information relating to an interest
9 rate and/or an exchange rate that can affect the calculation of the
10 amount of one or more of the payments made pursuant to the payment
11 schedule;
12 (b) reference data, including profile information about the parties to the
13 transaction; and
14 (c) market data, including market interest rates and/or currency exchange rates.
- 40 1 28. A method for converting between a first set of internal objects, representing the terms
2 of a transaction involving the exchange of financial instruments between a provider user of

5 the system and a member user of the system, and a transaction message employing a standard
3 format for representing the transaction, the method comprising the following steps:

- 6 (a) applying an XML mapping to convert between the first set of internal objects
5 and a Document Object Model tree; and
10 (b) applying an XSL stylesheet to convert between the Document Object Model
7 tree and the transaction message.

1 29. The method of claim 28 wherein the transaction message includes:

- 15 (a) transaction data describing the transaction, including:
2 (i) a type of the transaction;
3 (ii) a plurality of parties to the transaction, including the provider user and
20 the member user of the system;
5 (iii) a payment schedule pursuant to which the member user will make one
6 or more payments to the provider user; and
7 (iv) rate information which can include information relating to an interest
25 rate and/or an exchange rate that can affect the calculation of the
9 amount of one or more of the payments made pursuant to the payment
10 schedule;
11
30 (b) reference data, including profile information about the parties to the
12 transaction; and
13 (c) market data, including market interest rates and/or currency exchange rates.

35 30. The method of claim 28 wherein the standard format is an XML format.

1 31. The method of claim 28 wherein the first set of internal objects includes Java objects.

40 32. A method for converting between a transaction message employing a standard format
2 for representing the terms of a transaction involving the exchange of financial instruments
3 between a provider user of the system and a member user of the system, and a first set of
4 internal objects representing the transaction, the method comprising the following steps:

- 45 (a) applying an XSL stylesheet to convert between the transaction message and a
5 JavaScript program; and
6

5 (b) executing the JavaScript program in order to convert to the first set of internal
6 objects.

1 33. The method of claim 32 wherein the transaction message includes:

2 (a) transaction data describing the transaction, including:
3 (i) a type of the transaction;
4 (ii) a plurality of parties to the transaction, including the provider user and
5 the member user of the system;
6 (iii) a payment schedule pursuant to which the member user will make one
7 or more payments to the provider user; and
8 (iv) rate information which can include information relating to an interest
9 rate and/or an exchange rate that can affect the calculation of the
10 amount of one or more of the payments made pursuant to the payment
11 schedule;
12 (b) reference data, including profile information about the parties to the
13 transaction; and
14 (c) market data, including market interest rates and/or currency exchange rates.

1 34. The method of claim 32 wherein the standard format is an XML format.

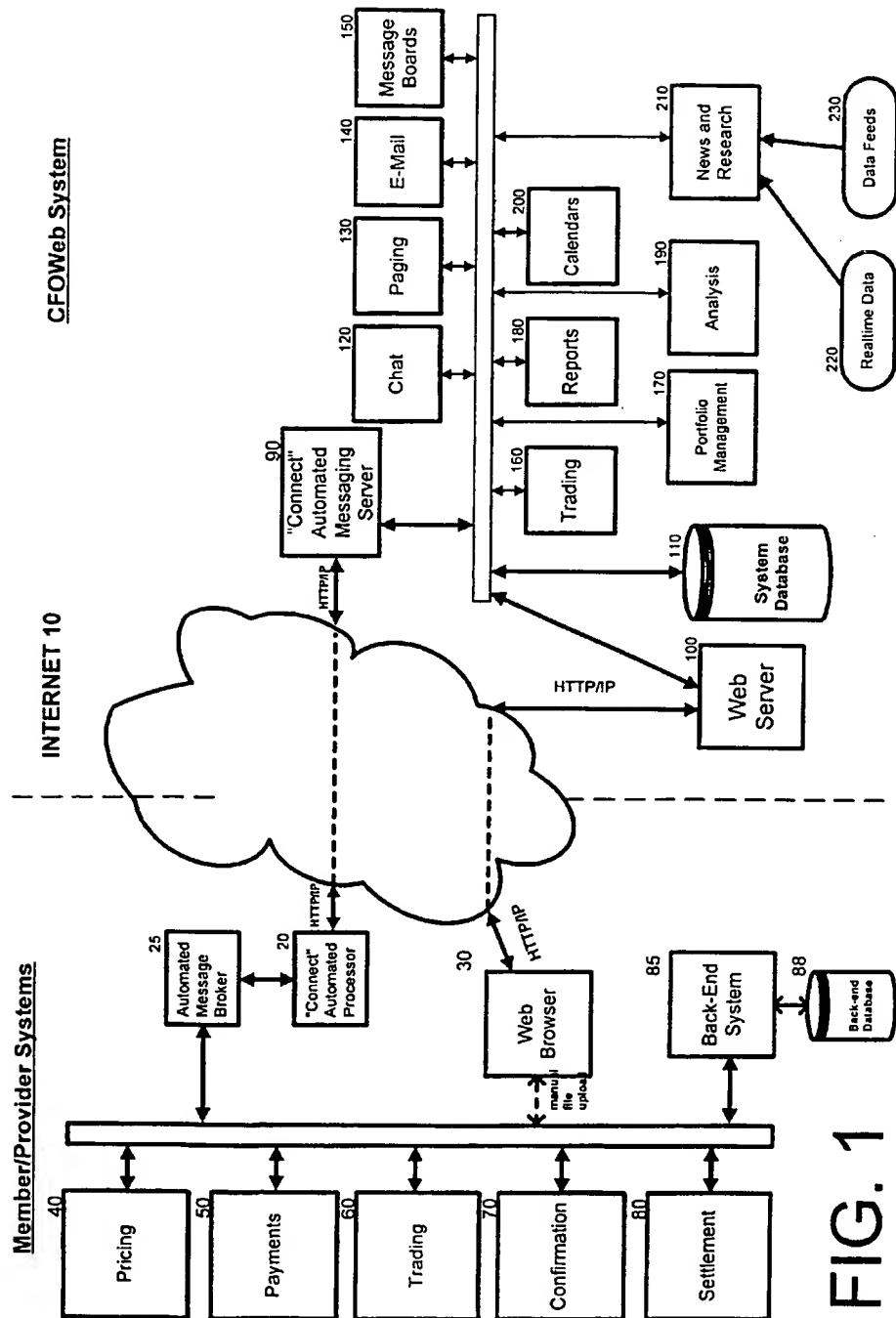
30 35. The method of claim 32 wherein the first set of internal objects includes Java objects.

35

40

45

50



一
EIG

Transaction Process Flow

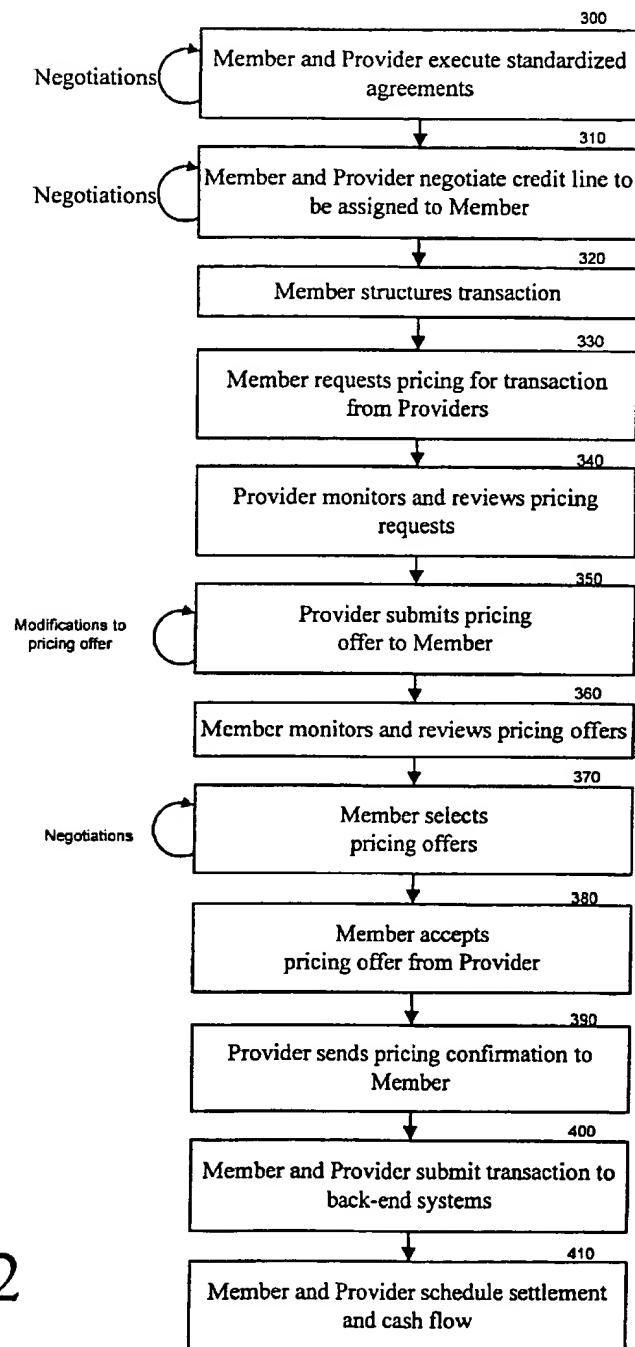


FIG. 2

FinXML Trade Element Structure

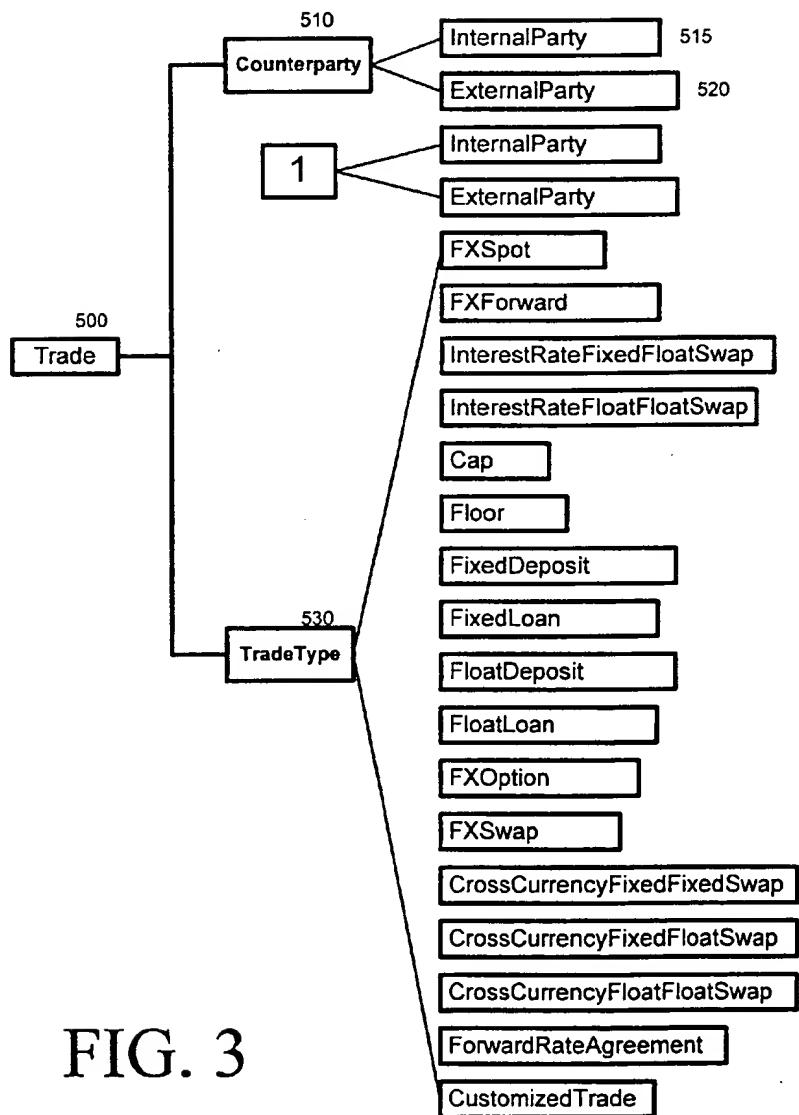


FIG. 3

FinXML External Party Definition

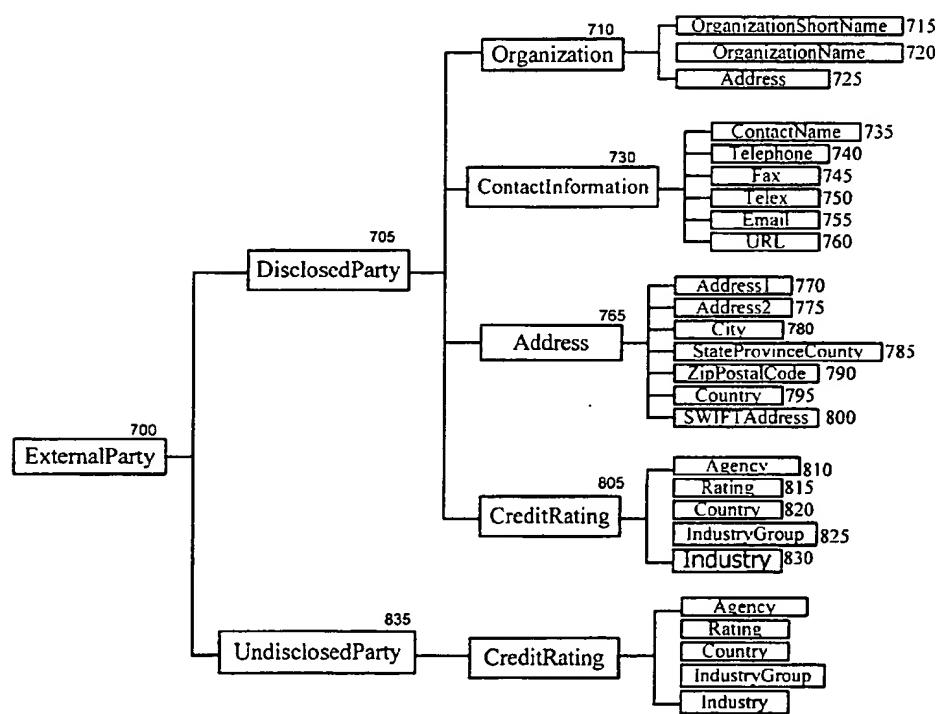


FIG. 4

FinXML Internal Party Definition

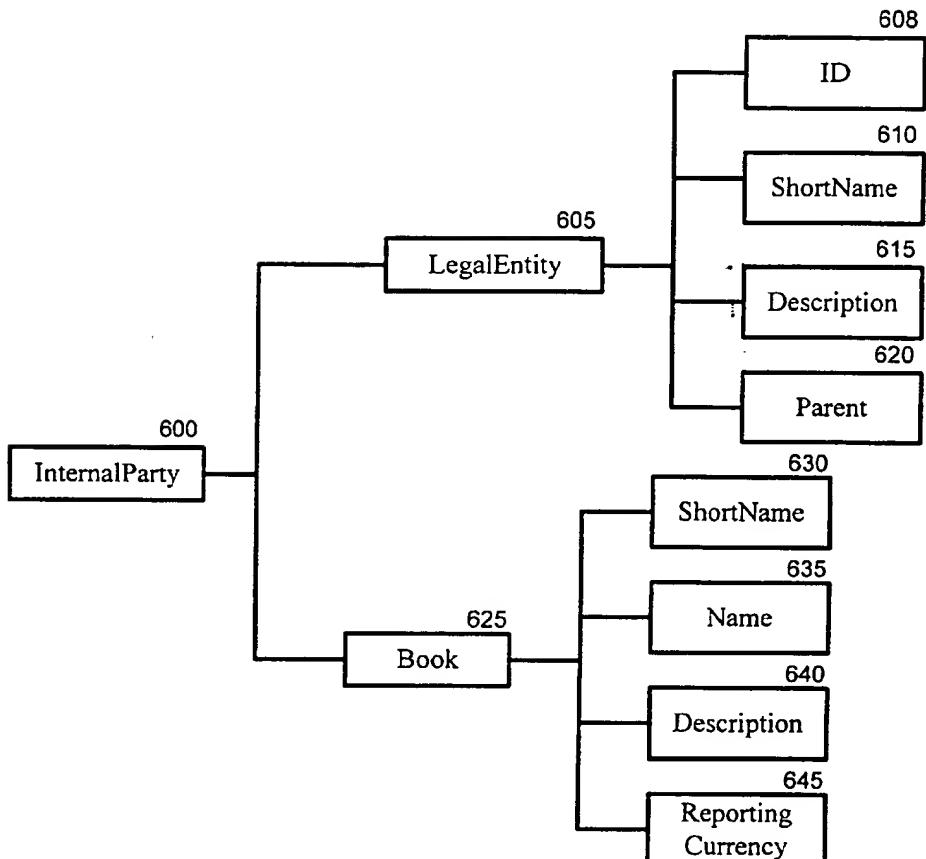


FIG. 5

FinXML Events Element Structure

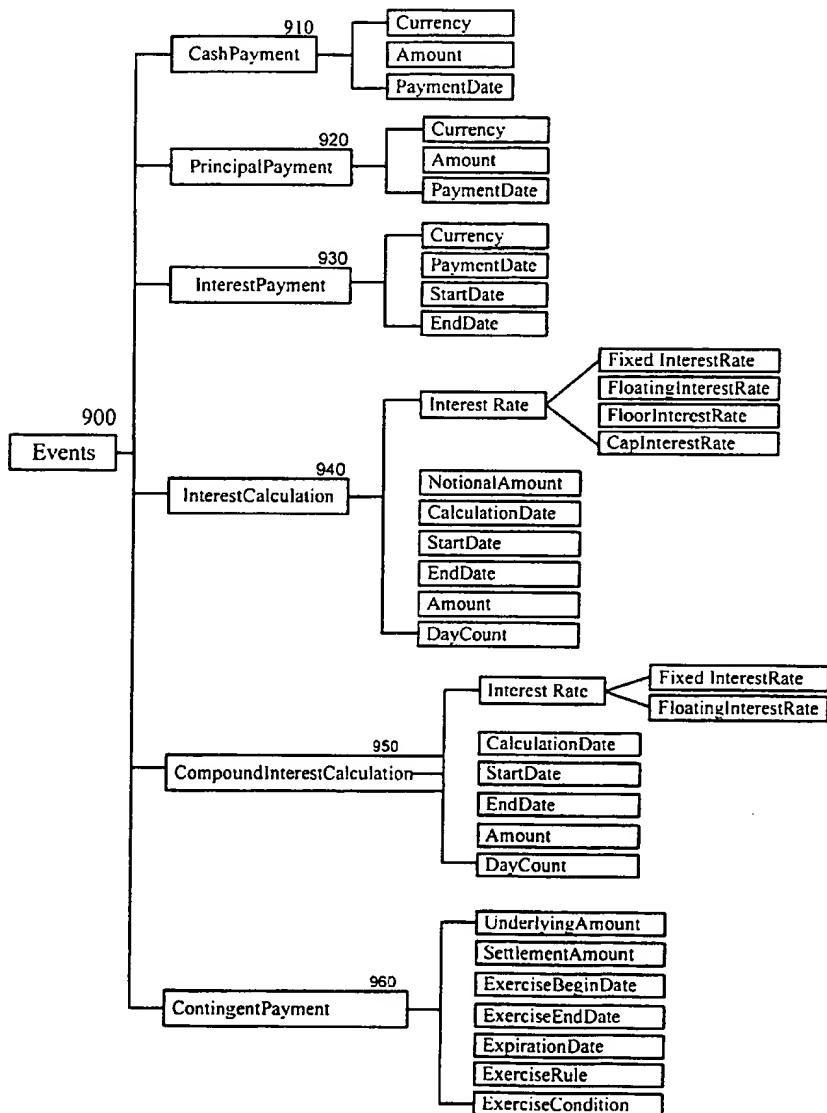


FIG. 6

Connect Automated Processor

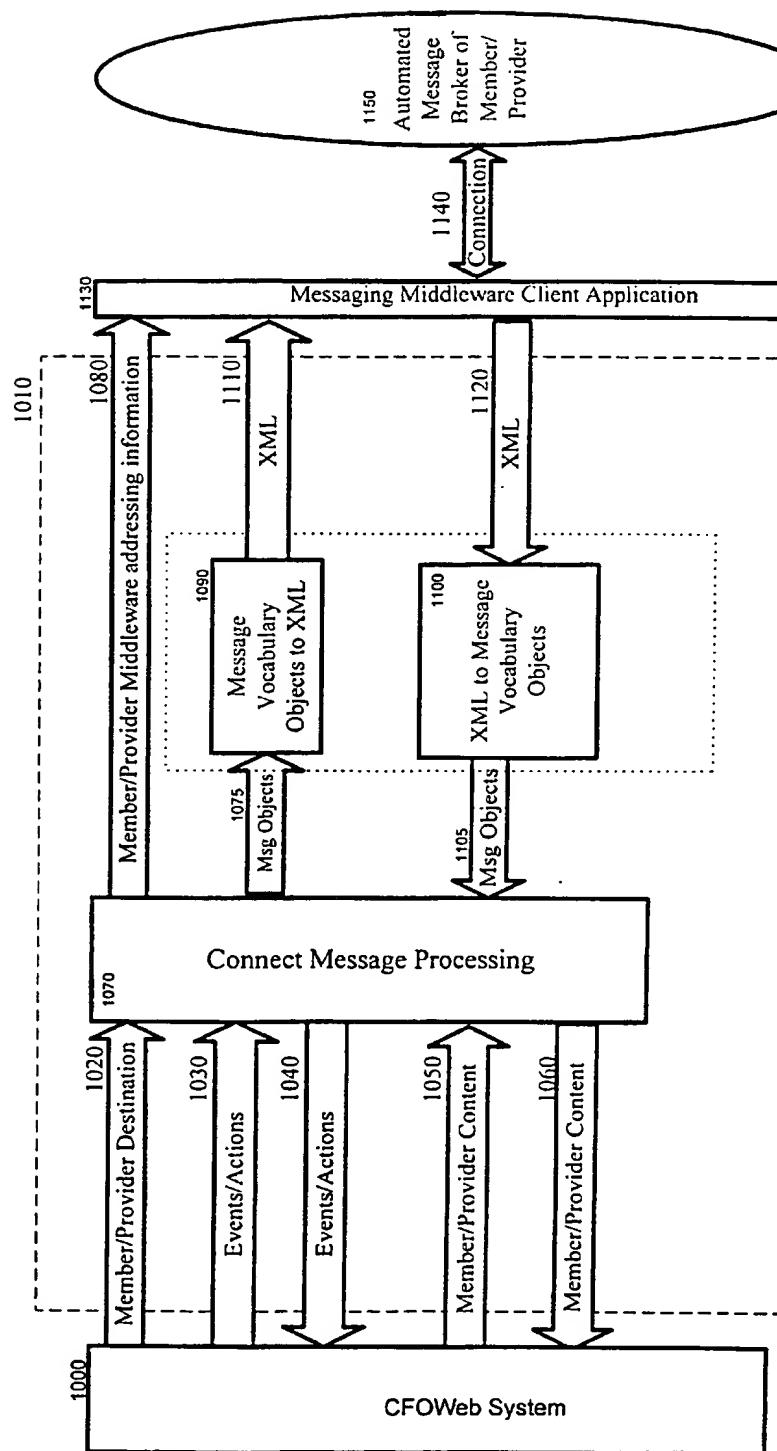


FIG. 7

Connect Architecture Overview

WO 00/77709

PCT/US00/16526

8 / 18

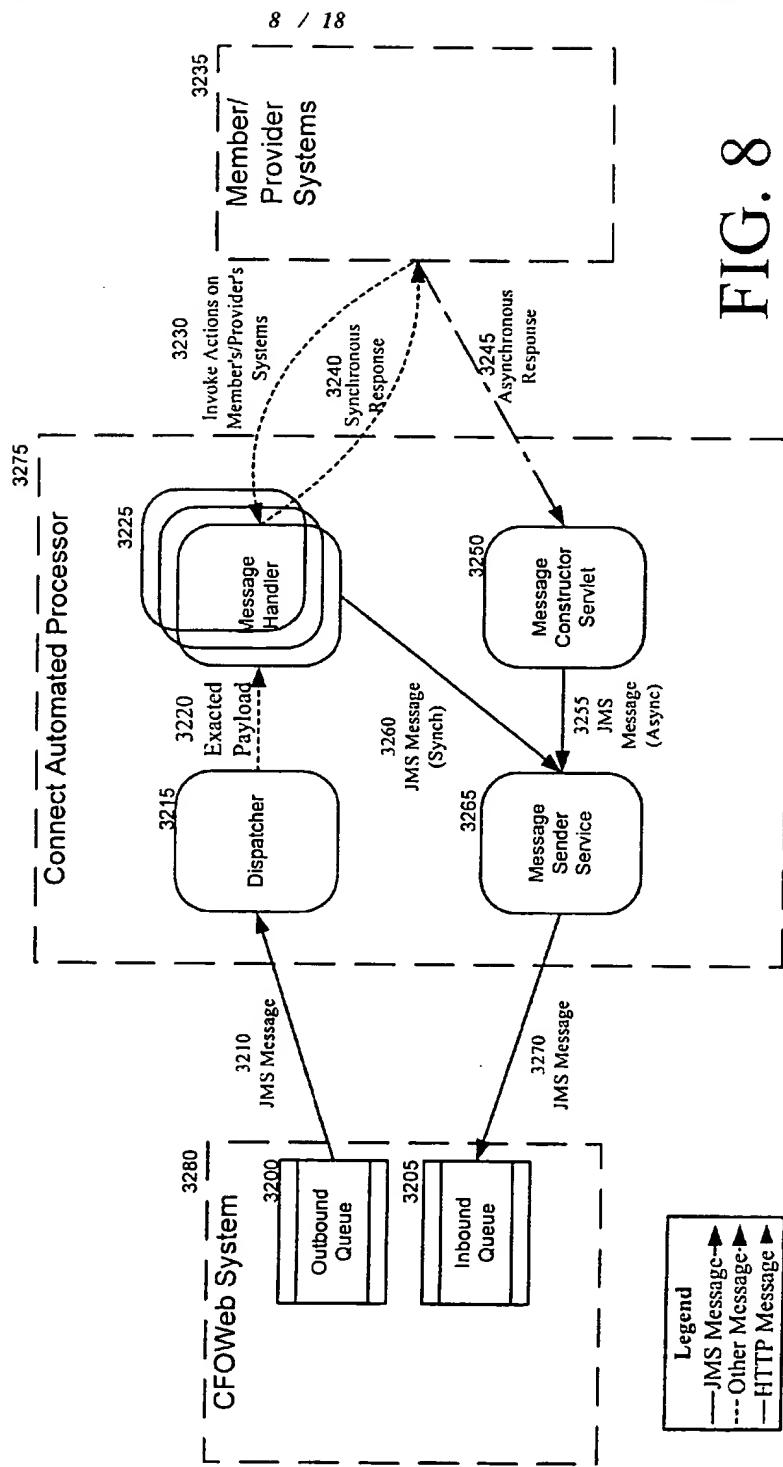


FIG. 8

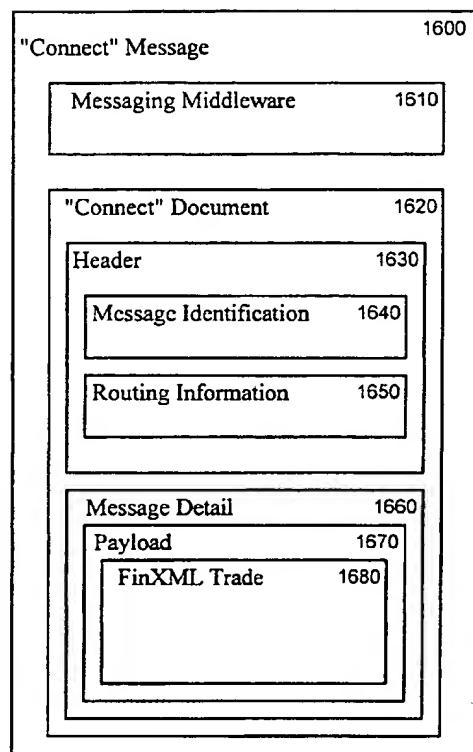
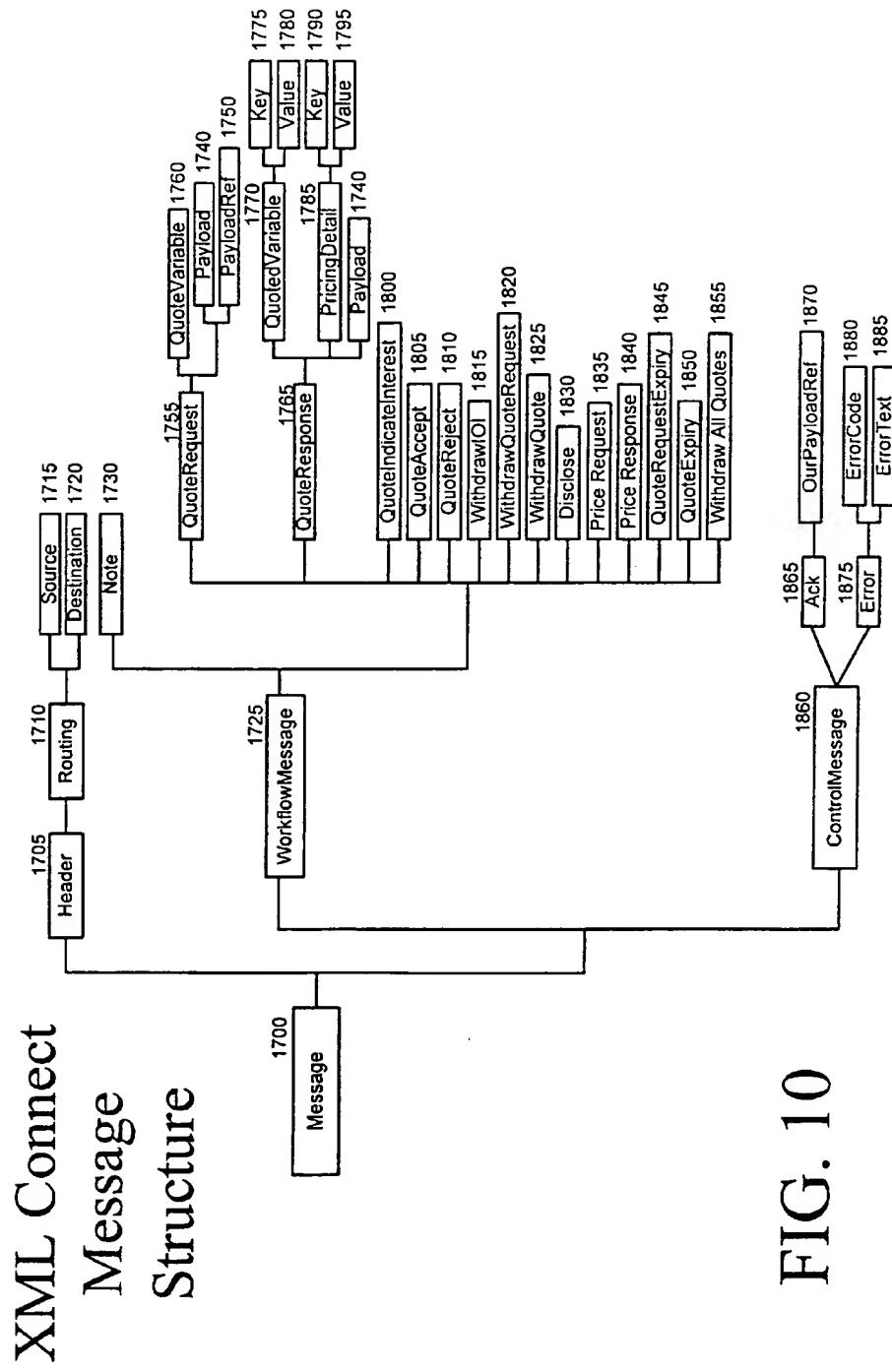


FIG. 9



Connect Message Flow - Automated Pricing (Synchronous)

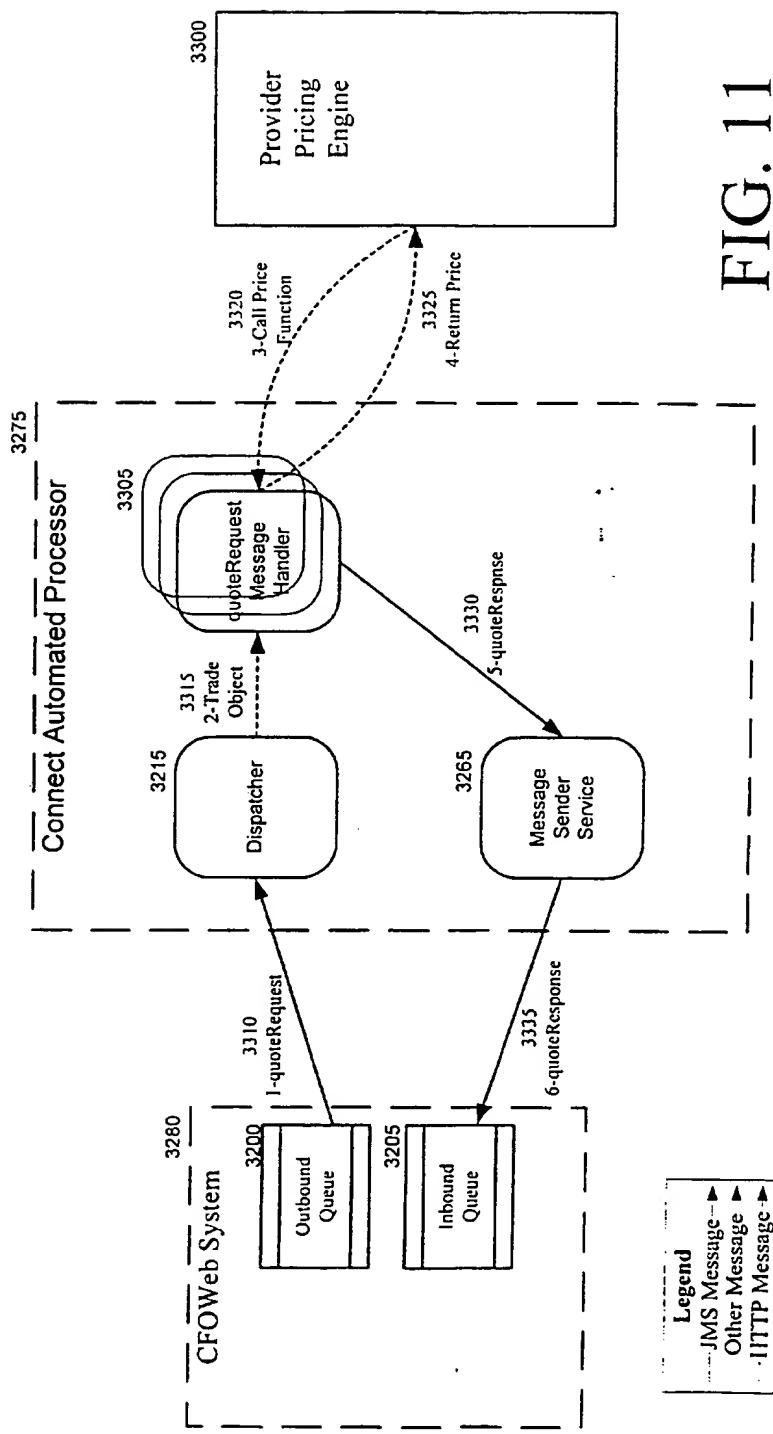


FIG. 11

Connect Message Flow - Automated Pricing (Asynchronous)

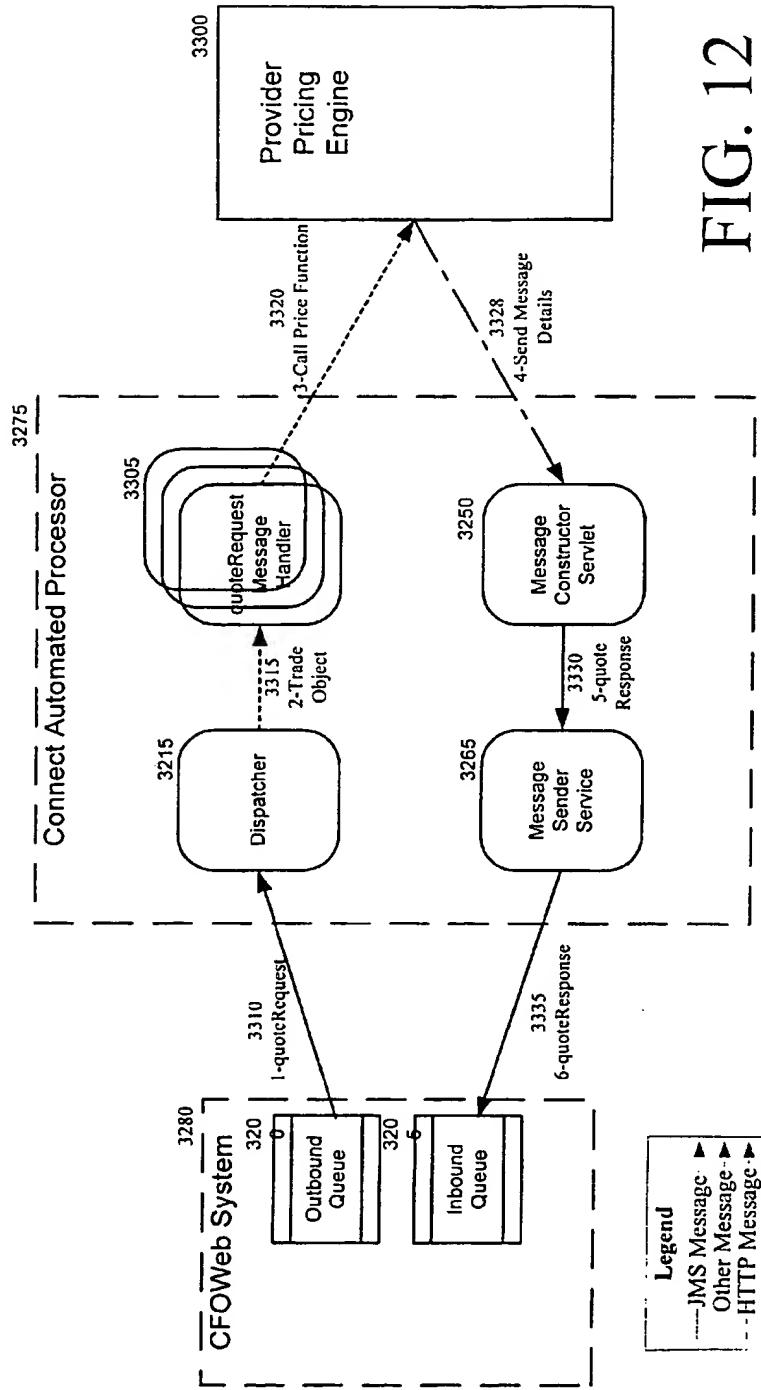


FIG. 12

Connect Message Flow - Semi-automated Pricing (Synchronous)

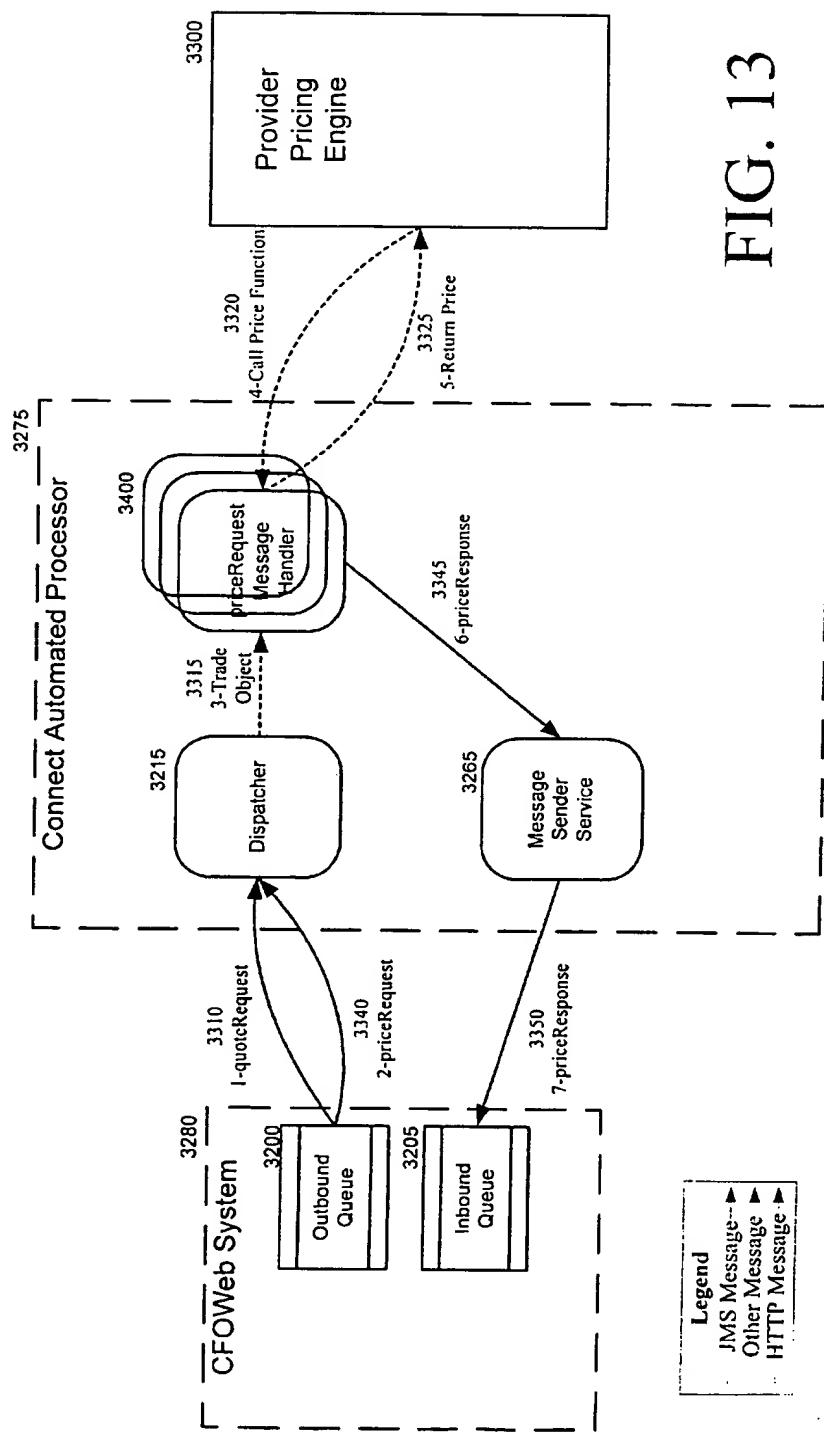


FIG. 13

Connect Message Flow - Deal Transmission (Asynchronous)

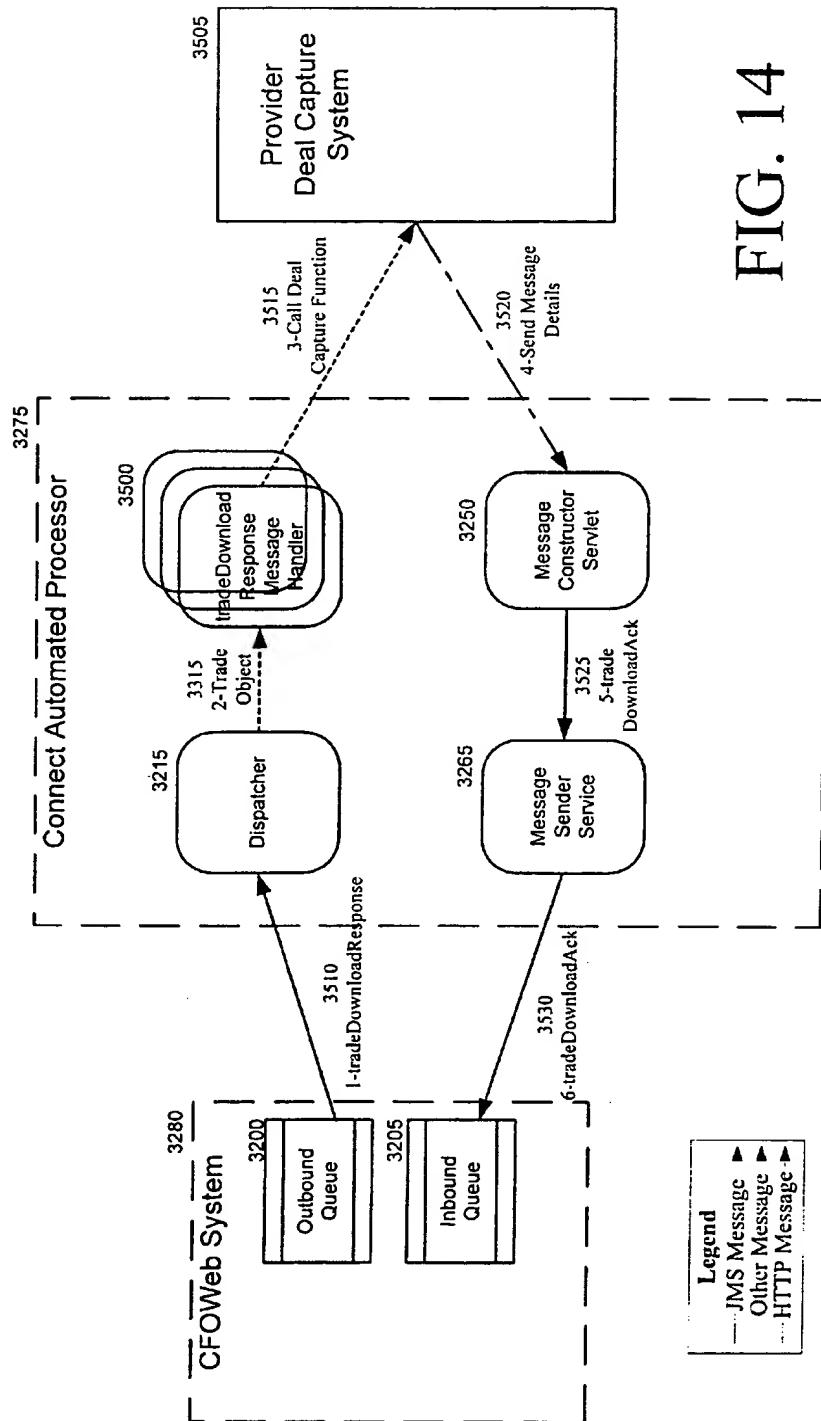


FIG. 14

Converting Objects to FinXML

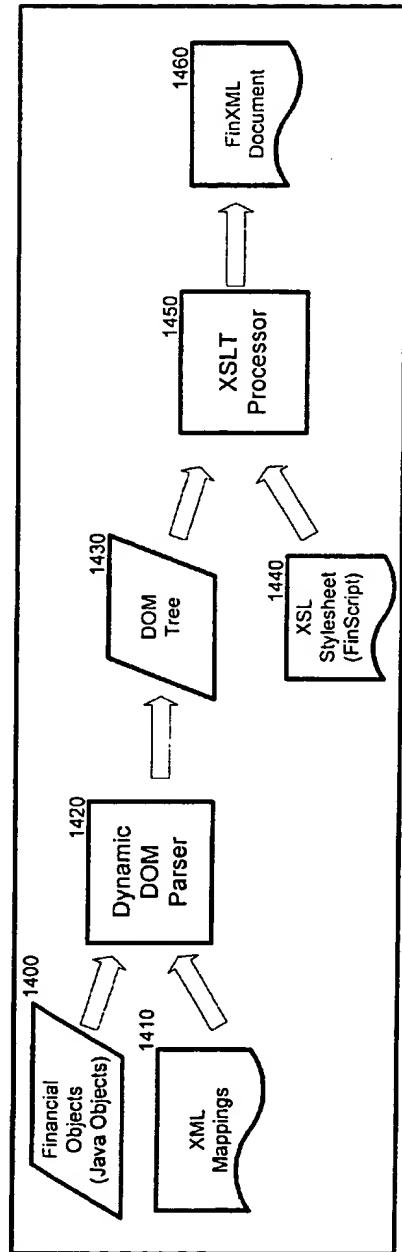


FIG. 15

1470

Messaging Client Sends
Financial Objects to Connect
Processor via API

1480

Connect Processor Identifies
Applicable XML Mapping

1490

Dynamic DOM Parser Parses
Financial Object and Applies
XML Mapping to Financial
Object

1500

Dynamic DOM Parser
Generates DOM Tree
Corresponding to Financial
Object

1510

Connect Processor Obtains
XSL Stylesheet (FinScript)

1520

XSLT Processor Applies XSL
Stylesheet to DOM Tree

1530

XSLT Processor Generates
FinXML Document

FIG. 16

Converting FinXML to Objects

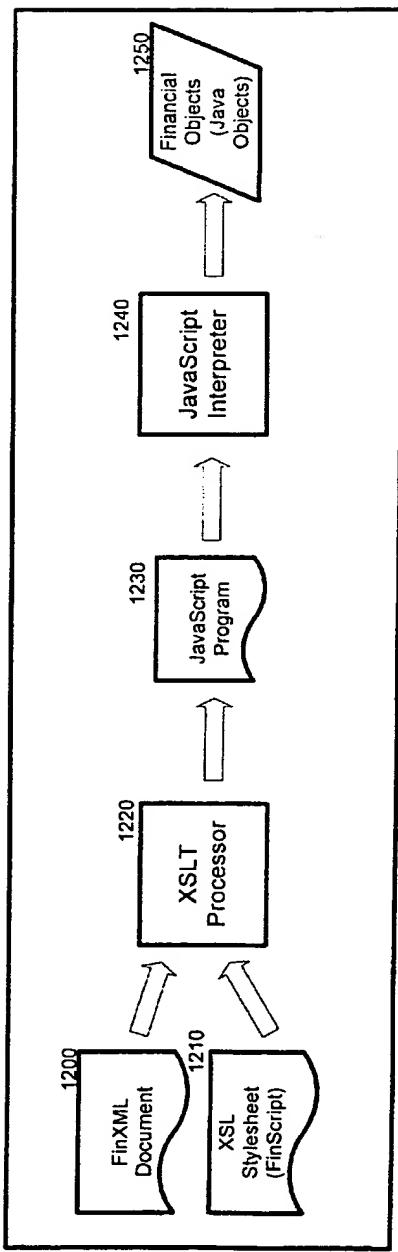


FIG. 17

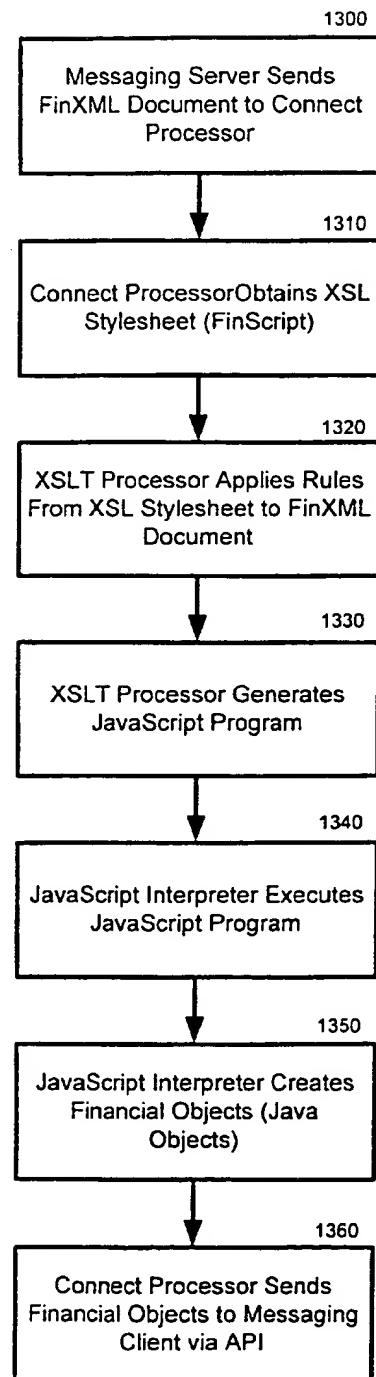


FIG. 18

INTERNATIONAL SEARCH REPORT

International application No. PCT/US00/16526

A. CLASSIFICATION OF SUBJECT MATTER		
IPC(7) : G06F 17/60 US CL : 705/37 According to International Patent Classification (IPC) or to both national classification and IPC		
B. FIELDS SEARCHED		
Minimum documentation searched (classification system followed by classification symbols) U.S. : 705/1, 35, 36, 37		
Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched SUN MICROSYSTEMS WEB PAGES, BANKING AND FINANCE TERMINOLOGY		
Electronic data base consulted during the international search (name of data base and, where practicable, search terms used) PROQUEST DATABASES Search Terms: foreign, exchange, internet, currency, trading		
C. DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 5,787,402 A (POTTER et al.) 28 July 1998, col. 4, lines 63-67; col. 5, lines 35-60; col. 7, lines 43-55.	1-25
Y, P	US 6,012,098 A (BAYEH et al.) 04 January 2000, col. 4, lines 1-22; col. 7, lines 38-41; col. 8, lines 64-66; col. 10, line 44.	25-35
Y, P	US 6,125,391 A (MELTZER et al.) 26 September 2000, col. 2, lines 32-54; col. 3, lines 31-38; col. 7, lines 34-36, 61-65; col. 24, lines 53-59.	1-35
A	J A V A S C R I P T F u n c t i o n a l D e s c r i p t i o n . http://home.netscape.com/eng/mozilla/3.0/handbook/javascript/getstart.htm . Pages 1-3.	1-35
<input checked="" type="checkbox"/> Further documents are listed in the continuation of Box C. <input type="checkbox"/> See patent family annex.		
* Special categories of cited documents: "A" document defining the general state of the art which is not considered to be of particular relevance "E" earlier document published on or after the international filing date "L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) "O" document referring to an oral disclosure, use, exhibition or other means "P" document published prior to the international filing date but later than the priority date claimed		
Date of the actual completion of the international search 03 OCTOBER 2000	Date of mailing of the international search report 27 NOV 2000	
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-3230	Authorized officer TARIQ HAFIZ <i>James R. Matthews</i> Telephone No. (703) 308-7808	

INTERNATIONAL SEARCH REPORT

International application No. PCT/US00/16526

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	JAVASERVLET Technology Product Description. http://java.sun.com/products/servlet . Pages 1-2.	1-35
A, P	US 5,913,202 A (MOTOYAMA) 15 June 1999, col. 2, lines 15-22.	1-25
Y, P	US 6,049,783 A (SEGAL et al.) 11 April 2000, col. 2, line 13-20, lines 55-56; col. 3, line 14; Fig. 1.	1-35
Y	US 5,497,317 A (HAWKINS et al.) 05 March 1996, col. 5, lines 14-15, lines 38-40, lines 64-66.	19-35
Y	US 5,630,127 A (MOORE et al.) 13 May 1997, col. 7, lines 16-19; col. 9, lines 65-67; col. 14, lines 14-19.	1-35
A	"Currency Management Corporation: CMC Hits \$20 Billion Traded Over Internet in Year." M2 Presswire. February 16, 1998. Pages 1-2.	1-35
A	HIGGINS, KIERAN. "Internet FX Trading Goes On-Line," Corporate Finance. September 1996. Pages 1-2.	1-35